Diplomarbeit

Konzeption eines webbasierten Managementsystems

Konzeption eines webbasierten Managementsystems

Diplomarbeit bei Xiranet Communications GmbH,
in Zusammenarbeit mit der
Technischen Universität Dresden

18. Dezember 2006

Peter Lieder

Betreuer: Dipl.-Inform. Mirko Benz, Dipl.-Inform. Stephan Groß
Verantwortlicher Hochschullehrer: Prof. Dr. habil. Alexander Schill

Fakultät Informatik
Institut für Systemarchitektur
Lehrstuhl für Rechnernetze

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit selbstständig und nur unter Verwendung der angegebenen Hilfsmittel geschrieben zu haben.

Dresden, 18. Dezember 2006 _____

Zusammenfassung

Mit der zunehmenden Integration von verteilten Systemen in der Wirtschaft sowie der Vernetzung von Geschäftsprozessen stiegen auch die Anforderungen an die Speichersysteme und deren Hersteller. Mit den steigenden Bedarf an Speicherkapazität wuchs auch das Verlangen nach Verfügbarkeit und Integrität der Daten sowie kompatiblen Lösungen für heterogene Netzwerkinfrastrukturen.

Im ersten Teil dieser Arbeit werden die Grundlagen für intelligente Speichesysteme betrachtet und mögliche Basistechnologien für eine Verwaltungssoftware untersucht. Weiterhin werden bestehende Ansätze für Verwaltungssysteme analysiert und bewertet. Mit den gewonnenen Erkenntnissen der Analysen wird ein Redesign des Xiranet Storage-Managementsystems durchgeführt. Die Konzeption umfasst eine inhaltliche und technische Überarbeitung des alten Managementsystems mit dem Ausblick auf einen neuen graphischen Ansatz. Die dabei parallel durchgeführte prototypische Implementierung ausgewählter Bereiche basiert auf dem Webentwicklungs-Framework Symfony und bietet viele neue Funktionalitäten an. Abschließend wird das neue Konzept bewertet und ein Ausblick auf das Aperi-Projekt gegeben.

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen von Storagesystemen **3**
2.1 Arten von Storagesytemen . . . 3
2.2 Aufbau von Xiranet Storagelösungen . . . 5

3 Evaluation von Basistechnologien **7**
3.1 Anforderungen an das Xiranet Managementsystem . . . 7
3.2 Apache Webserver und PHP . . . 9
3.3 Apache Tomcat mit Cocoon und XSP . . . 9
3.4 Zope und Python . . . 11
3.5 Java Client/Server-Anwendung . . . 11
3.6 Vergleich und Schlussfolgerung . . . 12

4 Web 2.0 Technologien **15**
4.1 Web Services APIs . . . 17
4.2 AJAX . . . 18
4.3 Abonnement-Dienste . . . 20
4.4 Soziale Software . . . 20

5 Analyse von Managementsystemen bestehender Storagelösungen **23**
5.1 Anforderungen an ein Managementsystem . . . 23
5.2 Bewertungskriterien . . . 23
5.3 Bewertung . . . 24
5.3.1 Cisco SN5428 . . . 24
5.3.2 Open-e iSCSI Enterprise . . . 25
5.3.3 Xyratex StorView . . . 27
5.3.4 LSI Logic MyStorage Management Software . . . 29
5.3.5 Intel® Storage System Software . . . 30
5.3.6 OpenFiler . . . 32
5.3.7 FreeNAS . . . 33
5.3.8 EuroNAS Premium . . . 34
5.3.9 Nimbus Halo Manager . . . 36
5.3.10 EqualLogic PS Serie . . . 37
5.4 Vergleich und Schlussfolgerung . . . 39

6 Konzeption des neuen Managementsystems 41
6.1 Anforderungen an das Xiranet Managementsystem 41
6.2 Analyse des alten Systems 42
6.3 Inhaltliche Überarbeitung 47
6.4 Technische Überarbeitung 50
6.4.1 Auswahl des passenden PHP-Frameworks 50
6.4.2 Das Symfony Framework 52
6.4.3 Umsetzung mit Symfony 60
6.5 Graphische Überarbeitung 64
6.6 Ausblick für zukünftige Erweiterungen 68
6.6.1 Assistenten 68
6.6.2 Weltkartenmetapher 68
6.6.3 Replikation und Backup 69

7 Zusammenfassung 71
7.1 Bewertung der Neukonzeption 71
7.2 Ausblick 71

Literaturverzeichnis 75
Literatur 76

Abbildungsverzeichnis

2.1 Zusammenspiel von SAN und NAS 6

4.1 Das Ablaufdiagramm der Kommunikation zwischen Client und Server 19

5.1 Die Managementoberfläche des SN5428 25
5.2 Die GUI bei der Verwaltung von iSCSI Targets 26
5.3 Die Startseite des StorView Managementsystems 28
5.4 Das MyStorage Benutzerfenster mit Bezeichnungen 29
5.5 Die Oberfläche der Managementkonsole bei der Konfiguration eines Clusters 31
5.6 Die Oberfläche des OpenFiler Webfrontend in der Konfigurationsseite für Volumen 32
5.7 Die FreeNAS Konfigurationsoberfläche bei der Veränderung der RAID Einstellungen 34
5.8 Die Startseite des EuroNAS Managementsystemes 35
5.9 Die Nimbus Oberfläche bei der Verwaltung eines RAID Sets 36
5.10 Die webbasierte Managementoberfläche von Equallogic 38

6.1 Die Systemverwaltung des Storage-Managementsystems 43
6.2 Menu für das Erzeugen von logischen Festplatten 44
6.3 Die Fehlerbeschreibung beim Anlegen eines logischen Volumen 45
6.5 Die Navigationsstruktur in einem Managementsystem 47
6.4 Strukturdiagramm des zukünftigen Storage-Managementsystems 48
6.6 Schematischer Zusammenhang des MVC-Modells 53
6.7 Das MVC-Modell von Symfony 54
6.8 Strukturelle Aufteilung des Grundlayouts 60
6.9 Die Interaktionsbeziehungen zwischen den Modulen 62
6.10 Baumnavigation und graphische Navigationselemente 65
6.11 Die graphische Darstellung des Inhaltes einer Zelle 66
6.12 Die Statusleiste mit teilweise ausgefahrenem Meldungsfenster 67
6.13 Das Einstellungsfenster für die Konfiguration der Statusleiste 67
6.14 Statusleiste mit vollständig ausgefahrenem Meldungsfenster 68
6.15 Beispiel einer möglichen Darstellung der Assistenten 69
6.16 Der graphische Prototype der Weltkartenmetapher 70

7.1 Die Architektur des Aperi Framework 73

Tabellenverzeichnis

3.1 Die Entscheidungskriterien im Überblick 13

4.1 Die Bedeutung von Web 2.0 16

5.1 Übersicht über die Eigenschaften der untersuchten Systeme 40

6.1 Die Entscheidungskriterien im Überblick 52
6.2 Aufwandsabschätzung für die Implementation 64

Listings

6.1 Verzeichnisbaum eines Symfony-Projekts 55
6.2 Verzeichnisbaum einer Symfony-Webanwendung 56
6.3 Verzeichnisbaum eines Symfony-Moduls 57

Abkürzungsverzeichnis

AJAX	Asynchronous JavaScript and XML
ATA/ATAPI	Advanced Technology Attachment with Packet Interface
Blog	Web Log
CIFS	Common Internet File System
CIM	Common Information Model
CRUD	Create, Retrieve, Update, Delete
CSS	Cascading Style Sheet
DAS	Direct Attached Storage
DOM	Document Object Model
FC	Fibre Channel
FTP	File Transfer Protocol
GB/s	Gigabyte pro Sekunde
GPL	General Public Licence
GUI	Graphical User Interface, graphische Benutzeroberfläche
HTML	Hyper Text Markup Language
IDE	Integrated Drive Electronics
iSCSI	Internet Small Computer System Interface
JSON	JavaScript Object Notation
LGPL	Lesser General Public Licence
MB	Megabyte
MVC	Model View Controller
NAS	Network Attached Storage
NFS	Network File System
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
REST	restructured Text
SAN	Storage Area Network
SAS	Server Attached Storage
SCSI	Small Computer System Interface
SDK	Software Development Kit
SMB	Server Message Protocol
SOAP	Simple Object Access Protocol
SSM	Storage System Module
TCO	Total costs of ownership
UDDI	Universal Description, Discovery and Integration
VM	Virtual Machine
W3C	World Wide Web Consortium
WBEM	Web Based Enterprise Management
WSDL	Web Service Description Language

XHTML ..... eXtensible HTML
XML ..... eXtensible Markup Language
XSLT ..... eXtensible Stylesheet Language Transformation

1 Einleitung

Mit der fortlaufenden Entwicklung der Computersysteme nimmt auch die Komplexität von Informationsverarbeitungssystemen zu. Die zu verarbeitenden Datenmengen werden größer und erfordern höhere Speicherkapazitäten und höchste Verfügbarkeit. Im Bereich der Speichernetzwerke erhalten neue Technologien wie iSCSI, InfiniBand und SAS den Einzug und tragen im Wesentlichen zu einer Performancesteigerung bei. Häufig ist ein Administrator für Systeme unterschiedlicher Aufgabenklassen und Hersteller verantwortlich und kann aufgrund der kurzen Einarbeitungszeit nicht das volle Potential der Lösung ausnutzen. Mit einem einfach und intuitiv zu bedienenden Verwaltungssystem kann dem Problem entgegengekommen werden. Der Erfolg sind geringere Managementkosten und eine effektivere Nutzung der Systeme.

Die Firma Xiranet Communications GmbH Dresden beschäftigt sich mit der Entwicklung von intelligenten Speichersystemen für den Einsatz in heterogenen Netzwerken. Dabei werden die Standardtechnologien TCP/IP und Ethernet in Verbindung mit dem relativ neuen Standard iSCSI genutzt, um eine kostengünstige und einfach zu verwaltende Speicherlösung in einer bestehenden Netzwerkinfrastruktur zu realisieren. Das Primärziel von Xiranet besteht in der Bereitstellung von technisch hochwertigen Speicherlösungen mit sicher aufeinander abgestimmten Komponenten. Der Administrator soll die Möglichkeit haben, eine Vielzahl an Einstellungen vorzunehmen ohne die Betriebssicherheit des Systems zu gefährden. Aus diesem Grund ist eine komfortable und übersichtliche Verwaltungsoberfläche notwendig, die in den bisherigen Systemen durch ein Webinterface verwirklicht wurden.

Das Ziel der Diplomarbeit ist es, die Technologien zur Realisierung eines webbasierten Managementsystems zu evaluieren und geeignet zu klassifizieren. Die beste Technologie ist in einer prototypischen Implementierung von ausgewählten Bereichen anzuwenden. Anschließend werden existierende Systeme zur Verwaltung von Speichersystemen untersucht, unter anderem mit den folgenden Kriterien: Menüführung, Umfang, Zusammenfassung von Arbeitsschritten und Visualisierung. Die daraus gewonnenen Erkenntnisse werden in die Konzeption für ein Redesign des Xiranet Managementsystems einfließen. Den Abschluss der Arbeit bilden eine Bewertung der Lösung im Vergleich mit anderen verfügbaren Systemen sowie ein Ausblick zu eventuellen Problemen.

Um die Konzeption des neuen Managementsystems durchführen zu können bedarf es dem Verständnis der technischen Grundlagen und der Analyse der Problemstellung. In Kapitel 2 werden mit den Grundlagen die Arten von Storagesystemen erläutert und spezielle Xiranet Lösungen vorgestellt. Mit der Evaluation von Basistechnologien in Kapitel 3 wird die optimale Basistechnologie für die Realisierung gesucht. Der dabei aufgestellte Kriterienkatalog basiert auf technischen Einschränkungen wie Ressourcenbedarf, Performance und Flexibilität. Das Kapitel 4 beschäftigt sich ausschließlich mit den Web2.0-Technologien, wobei aufgrund der späteren Implementierung der Schwerpunkt auf der AJAX-Technologie

liegt. Bei der Analyse von Managementsystemen bestehender Storagelösungen in Kapitel 5 wird ein Kriterienkatalog für eine weitere Bewertung aufgestellt und abgearbeitet, bei der die Eigenschaften der graphischen Benutzungsschnittstelle im Mittelpunkt stehen. Das Kapitel 6 nutzt die zuvor gewonnenen Erkenntnisse für die Konzeption des neuen Managementsystems und den Ausblick auf zukünftige Erweiterungen. Abschließend wird die Neukonzeption bewertet und mit einem Ausblick abgeschlossen.

2 Grundlagen von Storagesystemen

Storagesysteme sind leistungsstarke, autonome Rechnersysteme. Sie bieten eine ausfallsichere Speicherumgebung für andere Computer in einem Netzwerk an. In Abhängigkeit von den Anforderungen des Einsatzgebietes bieten diese Systeme angepasste Dienste für die Kommunikation an. In kleinen Firmennetzwerken können so genannte stand-alone-Lösungen ein kostengünstiger Einstieg sein. Oder sie erweitern ein bereits existierendes Speichernetzwerk um eine weitere Komponente und erhöhen damit nicht nur die Kapazität, sondern auch die Flexibilität des gesamten Systems. Der Zusammenhang und die Bedeutung der dabei eingesetzten Technologien wird im folgenden Abschnitt genauer betrachtet.

2.1 Arten von Storagesytemen

Die **NAS** (*Network Attached Storage*) Systeme sind Massenspeichereinheiten, die an ein Ethernet Netzwerk angeschlossen werden. Sie dienen der Erweiterung der Speicherkapazität und übernehmen die Aufgaben eines dedizierten Dateiservers. Die Kernfunktion von NAS-Systemen sind die dateibasierenden Dienste wie NFS oder SMB/CIFS. Jeder Zugriff auf das NAS-System ist ein Zugriff auf ein existierendes Dateisystem und wird vom System selbst verwaltet. Im Vergleich zu den speziell für Speichernetzwerke entwickelten Übertragungstechnologien hat das hier verwendete Ethernet geringere Rahmengrößen und einen höheren Protokoll-Overhead. Dadurch wird das Netzwerk stark ausgelastet und ein schneller Datenzugriff ist nicht möglich. Zum Einsatz von eigenen Festplatten als Speichermedien lassen sich auch andere Speichereinheiten über ein SAN integrieren.[1]

Ein **SAN** (*Storage Area Network*) ist eine Netzwerktechnologie speziell für den Datenaustausch zwischen Servern und deren Speicherressourcen. Der Datenaustausch findet dabei blockbasiert statt und nutzt Fibre Channel oder Ethernet als Übertragungsmedium. Es muss dafür kein Dateisystem zur Verfügung gestellt werden, um einen Zugriff zu ermöglichen und Zugriffsrechte zu verwalten. Meist wird ein SCSI-Kommunikationsprotokoll genutzt, welches auf Fibre Channel (FC) oder iSCSI aufsetzt und auf Grund seines Durchsatzes geeigneter ist als herkömmliche Protokolle.[1]

Das **DAS** (*Direct Attached Storage*) bezeichnet das Prinzip der Direktverbindung von Massenspeicher mit dem Server. Die Speichermedien können interne oder externe Festplatten sein, die über die DAS-Protokolle ATA/ATAPI und SCSI kommunizieren können. Diese sehr einfache Speichertopologie wird in seltenen Fällen auch als *Server Attached Storage* bezeichnet.

Die weiteren Begriffe sollen nur kurz erläutert werden:

- **SMB**
  Das *Server Message Protocol* ist ein Kommunikationsprotokoll für ein weites Feld von Netzwerkdiensten. Es wird sowohl von kommerziellen Systemen wie Microsoft LAN-Manager oder IBM LAN-Server verwendet, als auch von der freien Software Samba.

- **NFS/CIFS**
  Der *Network File System* Dienst ermöglicht den dateibasierten Datenaustausch unter Unix und Linux. Ein ähnlicher Dienst, der 1996 von Microsoft eingeführt wurde, ist **CIFS**, das *Common Internet File System*. Es ist eine Erweiterung des bekannten SMB Protokolls. Es enthält neben der Datei- und Druckerfreigabe auch den Windows RPC und den NT-Domänendienst.

- **Fibre Channel**
  Fibre Channel ist eine eigenständige Netzwerktechnologie parallel zu „TCP/IP over Ethernet“. Es wurde speziell für den blockorientierten Datentransfer optimiert und findet deshalb hauptsächlich im SAN Bereich seinen Einsatz. Es werden ebenfalls SCSI-Steuerkommandos für die Übertragung verwendet, das Übertragungsmedium ist jedoch kein lokal beschränktes Bussystem, sondern ein beliebig verknüpfbarer Lichtwellenleiter. Fibre Channel hat eine Nutzdatenauslastung von über 90%, die bei Ethernet nur 20-60% beträgt. Dies ist ein entscheidender Vorteil für den Einsatz als Transportmedium für den Transfer für Massendaten über weite Strecken mit hoher Übertragungsgeschwindigkeit. Die SAN Architektur ist der LAN Architektur sehr ähnlich. Die Hosts, die als Server oder Storage-Elemente agieren können, werden durch ihren Hostbusadapter mit den Koppelelementen des Netzwerkes verbunden. Es existieren zwei Topologien für die Verbindung. Zum einen die **Loop** Topologie, bei der alle Geräte in einem Ring miteinander verbunden sind und die gesendeten Daten durch jedes Gerät „durchgeschleift“ werden. Zum anderen existiert die **Fabric** Topologie, bei der alle Geräte durch ein Switch verbunden sind und in Echtzeit durchgeschaltet werden. Da sich bei der Loop Topologie alle Geräte die Bandbreite teilen müssen, wird diese Topologie ab 5 Geräten ineffizient und deshalb nur selten eingesetzt.[1, 2]

- **InfiniBand**
  InfiniBand ist eine Spezifikation zur seriellen Hochgeschwindigkeitsübertragung von Daten, die früher auch unter dem Namen *System I/O* bekannt war. Große Firmen wie Intel, Microsoft und Sun beteiligten sich an deren Entwicklungsprozess und schufen die Technologie, die heute hauptsächlich zum Zweck der Clusterverbindung eingesetzt wird. Die Datenübertragung erfolgt über einen bidirektionalen seriellen Bus, der durch die geringe Latenz Datenraten bis zu 2,5 GB/s pro Kanal in jede Richtung erreicht. Die geringe Latenz wird durch die Auslagerung des Protokollstacks auf die Netzwerkhardware ermöglicht. Standardmäßig werden pro Verbindung vier oder acht Kanäle gebündelt und man erreicht Datenraten von 10 bis 20 GB/s. Als Übertragungsmedium wird für Strecken von bis zu 17 Metern Twinax Kupferkabel verwendet. Bei größeren Entfernungen bis zu 100 Meter können auch Lichtwellenleiter eingesetzt

werden, bei denen die InfiniBand Kanäle auf einzelnen Faserpaaren übertragen werden. Die Kopplung mehrerer Rechnerknoten erfolgt über spezielle Switches. Auch eine Verbindung mit existierenden IP Netzwerken kann durch das InfiniBand Protokoll realisiert werden, da die Adressierung mittels 128Bit Adressen außerhalb des Subnetzes durchgeführt wird. Mehr Informationen dazu sind in [3] enthalten.

- **iSCSI**
  Das *Internet Small Computer System Interface* ist ein Protokoll für die Übertragung von Speicherbefehlen über das IP Protokoll. Dabei werden die SCSI-Daten in TCP/IP-Pakete verpackt und über das Netzwerk transportiert. Ein spezieller iSCSI-Router, der fähig ist, die SCSI-Kommandos in den Paketen zu lesen, leitet die Pakete mit Hilfe von Mapping-Tabellen an das richtige Zielsystem weiter. Dadurch wird eine virtuelle Ende-zu-Ende-Verbindung erzeugt. Obwohl iSCSI langsamer als FC ist, sind die Anschaffungs- und Unterhaltungskosten(TCO) geringer. Zudem ist es flexibel in LAN und WAN einsetzbar mit einer annähernd guten Performance wie in FC Netzwerken. [1]

- **SAS**
  Das *Serial Attached SCSI* ist die Weiterentwicklung des parallelen UltraWideII SCSI Bussystems. Bei jedem Zugriff wird eine serielle Punkt-zu-Punkt-Verbindung zwischen Festplatte und Controller aufgebaut, die Übertragungsgeschwindigkeiten von drei bis zwölf Gigabit pro Sekunde erreicht. Für die Übertragung werden ebenfalls wieder SCSI-Speicherbefehle verwendet. Das physische Interface ist kompatibel mit dem SATA Standard, so dass auch SATA Festplatten an einem SAS-System betrieben werden können. Zusätzlich steigert das Dual-Porting die Verfügbarkeit der Speichermedien, indem jede Festplatte zwei Ports für den Anschluss an einen Controller besitzt.

- **JBOD**
  Just a Bench of Disks ist die Bezeichnung für eine Bündelung von Festplatten in einem Gehäuse, die über ein Bussystem mit dem Controller verbunden sind, jedoch nicht in einem RAID-Verbund integriert sind. Jede dieser Festplatten kann separat angesteuert und vom Controller verwaltet werden.

In der Praxis existieren auch hybride SAN und NAS Lösungen. Die NAS-Systeme besitzen SAN Schnittstellen, damit sie als Gateway zwischen LAN und SAN Netzwerken eingesetzt werden können. Um so mehr Dienste ein solches Speichersystem zur Verfügung stellt, desto flexibler ist es in seinem Einsatzbereich. In Abbildung 2.1 wird ein einfaches Szenario dargestellt, was die Begriffe NAS und SAN eindeutig differenzieren soll.

2.2 Aufbau von Xiranet Storagelösungen

Derzeit bietet Xiranet iSCSI Speichersysteme verschiedener Größen und einen Storage Router an. Das Speichersystem findet Einsatz in iSCSI basierten Storage Area Networks. Das XAS500, das kleinste Speichersystem im Angebot, ist ein eigenständiges System. Als Speichermedien können 15 hotplugin-fähige SATA oder SCSI Festplatten verwendet werden, die zu einer maximalen Kapazität von bis zu 7,5TB kombiniert werden können. Für die Netzwerkanbindung existieren momentan Gigabit Ethernet, 10GB Ethernet und Native

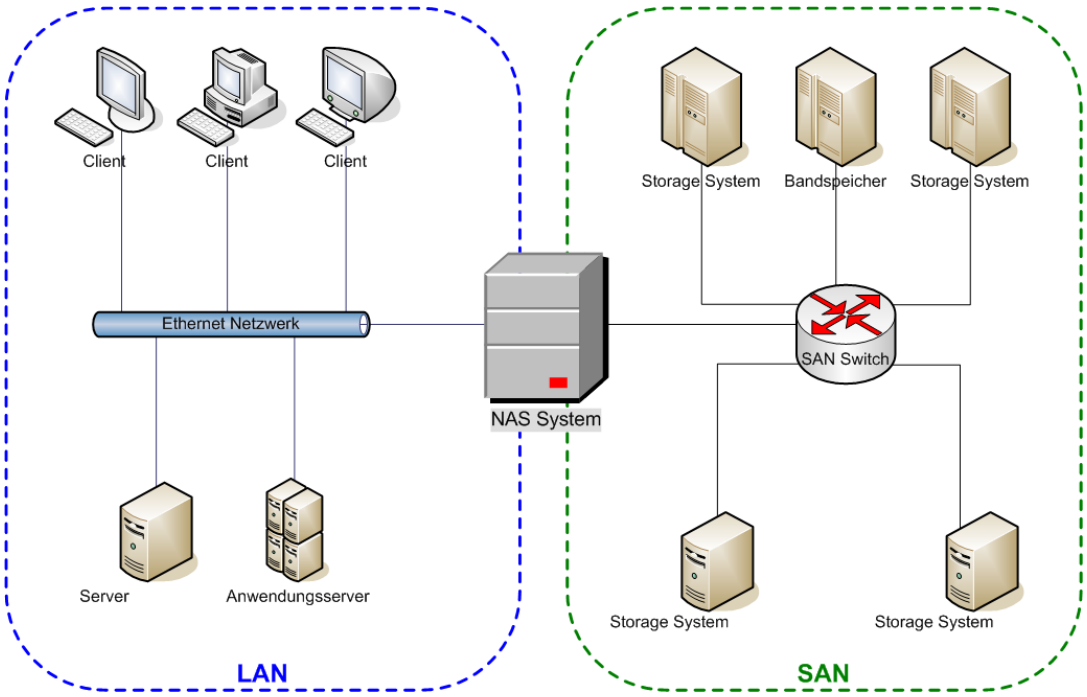

Abbildung 2.1: Zusammenspiel von SAN und NAS

InfiniBand. Zudem bietet es Unterstützung für verschiedene RAID Levels, IPSec Kommunikation, dynamische Volumenerweiterung und Snapshotverwaltung. Dieses Gerät ist also aufgrund seiner technischen Eigenschaften ein reines SAN Speichersystem. Auf einer höheren Stufe arbeitet das XAS1000 Enterprise Storagesystem. Es kann modular erweitert werden und besteht in der Grundkonfiguration aus einem Controller und einem JBOD. Jeder Controller hat vier SAS Ports, so dass in diesem Fall alle vier Ports mit einem JBOD verbunden werden. Im Falle einer Erweiterung auf zwei oder mehr JBODs werden immer zwei Ports für die Verbindung von Controller zu JBOD verwendet. Es können maximal zwei Controller eingesetzt werden, die ebenfalls über zwei Ports miteinander verbunden werden. Bis zu 16 JBODs können in einem solchen System angesteuert werden. Bei einer maximalen Anzahl von 12 Festplatten pro JBOD mit maximal 500GB kann eine Gesamtkapazität von 120 Terrabyte erreicht werden. Weiterhin gehören Multipathing, Clustering, Replikation und Snapshots zu den Eigenschaften, die ein solches System ausfallsicher und flexibel machen.

Der Storage Router ist multiprotokollfähig und hat die Aufgabe, existierende SCSI oder Fibre Channel Speichersysteme in das IP SAN zu Integrieren. Er hat also die Funktionalität eines SAN Gateways zwischen iSCSI Gigabit Ethernet und Fibre Channel. Auch die direkte Kopplung von Fibre Channel SANs wird unterstützt, was die Spiegelung von FC Volumen über das Internet ermöglicht. Zusätzlich bietet er auch die Verwaltung von RAID und logischen Volumen an, wodurch eine bessere Administration und Speicherauslastung möglich ist. Mit den redundanten Systemkomponenten sorgt der Storage Router für eine hohe Betriebssicherheit bei niedrigen Betriebskosten.

3 Evaluation von Basistechnologien für die Realisierung des Managementsystems

Dieses Kapitel enthält die Ergebnisse der vorbereitenden Analysephase dieser Arbeit. Dabei wurden die derzeit existierenden Möglichkeiten einer technischen Umsetzung nach bestimmten Kriterien untersucht. In Abschnitt 3.1 werden die Kriterien aufgestellt, nach denen die folgenden fünf Abschnitte die Technologien bewerten. Abschließend werden die Ergebnisse in Abschnitt 3.6 zusammengefasst.

3.1 Anforderungen an das Xiranet Managementsystem

Das zu entwickelnde Managementsystem ist stark abhängig von der eingesetzten Hardware. Aus welchen Kriterien die Anforderungen bestehen, sollen die folgenden Punkte näher erläutern:

- **Ressourcenbedarf**
  In der Konzeption des Storagesystems wird sehr viel Wert auf eine hohe Performance gelegt. Mit der Umsetzung dieses Zieles hat sich die Firma Xiranet ihre Marktfähigkeit gesichert. Deshalb darf das Managementsystem der jeweiligen Storagelösung keinen Performanceverlust bewirken. Es sollte nicht mehr als 10 Prozent des verfügbaren Hauptspeichers verwendet werden. Die gesamte Anwendung sollte nicht größer als 15 Megabyte sein, da bereits das Betriebssystem ca. 45 Megabyte Festplattenkapazität beansprucht. Auch die CPU-Belastung sollte nicht wesentlich ins Gewicht fallen, also eine Grenze von maximal 15 Prozent nicht übersteigen.

- **Ausführungsgeschwindigkeit**
  Es existieren Verwaltungsoperationen, die in ihrer Ausführung viel Zeit benötigen, wie beispielsweise das Anlegen eines neuen RAID Arrays. Der Nutzer des Managementsystems sollte die Möglichkeit haben, parallel weiterarbeiten zu können, ohne auf das Ergebnis des ersten Abarbeitungsprozesses warten zu müssen. Es muss also eine asynchrone Kommunikation direkt oder indirekt ermöglicht werden.

- **Netzwerkbelastung**
  In einem lokalen Netzwerk fällt die Netzwerkbelastung für administrative Aufgaben sehr gering aus. Allerdings soll das Storagesystem unabhängig vom Einsatzort administriert werden können, beispielsweise auch über ein Wide Area Network. Deshalb soll die zu übertragenden Datenmengen so gering wie möglich gehalten werden.

- **Implementationsaufwand**
  Bei der Suche nach der besten Technologie ist es auch interessant, wie viel Aufwand

für eine Implementierung aufgebracht werden muss. In der Praxis ist es meistens lohnenswert, mehr Aufwand in die Konzeption zu stecken und dafür dann eine flexiblere Lösung zu haben, die zu einem späteren Zeitpunkt beliebig erweitert werden kann. Dieses Kriterium ist also nur sekundär wichtig und sollte im Zusammenhang mit den anderen Kriterien in die Gesamtbewertung eingehen.

- **Schutz des Quelltextes**
  Prinzipiell sollen die Quelltexte des Managementsystems nicht mit der Hardware an den Kunden ausgeliefert werden, um eine Fremdvermarktung durch Dritte zu verhindern. Deshalb ist es von großer Bedeutung die Anwendung in eine nicht reproduzierbare Form zu überführen, ohne dabei die Lauffähigkeit negativ zu beeinflussen.

- **Lizenzbedingungen**
  Für die Veröffentlichung von eigenen Softwareprojekten in Zusammenhang mit unterstützenden bzw. verlinkten Teilen von Software dürfen die damit verbundenen Lizenzbedingungen nicht außer Acht gelassen werden. Die folgenden Lizenztypen sind dabei ausschlaggebend:

  - **GPL** (General Public License)
    Ein Programm, das mit der GPL lizenziert ist, darf ohne Einschränkungen genutzt, kostenlos (mit Quellcode) verteilt und verändert werden und veränderte oder unveränderte Programme dürfen unter Offenlegung des Quellcodes vertrieben werden. Diese Lizenz stellt damit jedoch die Bedingung, dass ein Projekt, welches eine unter der GPL veröffentlichte Bibliothek oder Programmteile nutzt, ebenfalls mit der GPL verbreitet werden muss.
  - **LGPL** (Lesser General Public License)
    Diese Form ist eine Auflockerung der GPL. Sie soll dem Entwickler erlauben, eine dynamische Verlinkung auf Bibliotheken mit LGPL durchzuführen, ohne seine eigene Software unter dieser Lizenz veröffentlichen zu müssen. Häufig wird diese Lizenz für die Programmbibliotheken einer Programmiersprache verwendet, wie zum Beispiel die Implementierung der Standard C Library *glibc*. Änderungen an diesen Softwareteilen müssen aber trotzdem noch mit ihren Quelltexten veröffentlicht werden.
  - **MIT**
    Die vom *Massachusetts Institute of Technology* eingeführte Lizenz für Software wird häufig von Entwicklergruppen und kleinen Softwareherstellern angewendet. Die Lizenz erlaubt das unentgeltliche Nutzen, Verändern und Verkaufen dieser Software, solange in jedem Teil der Software ein Urheberrechtsvermerk aufgeführt wird. Bekannte Beispiele für solche Software sind das XWindow-System X11 und der Expat Parser von James Clark.
  - **Speziell definierte Lizenzen**
    Wenn eine eigenständige Lizenz vom Hersteller von Software oder Bibliotheken erstellt wurde, dann sollte diese unbedingt beachtet werden. Eine Vernachlässigung der Lizenzrechte könnte schwerwiegende Folgen für das gesamte Projekt haben. Da eine genaue Aufschlüsselung der Lizenzinhalte bei der späteren Betrachtung sehr umfangreich sein kann, soll nur überprüft werden, ob die Technologie für den Anwendungszweck des Managementsystems nutzbar ist, bezie-

hungsweise aus welchen lizenzrechtlichen Gründen es nicht nutzbar ist. Auf eine vollständige Darstellung der Lizenzbedingungen wird jedoch verzichtet.

- **Nutzerfreundlichkeit**
  Grundlegend wurde für die Umsetzung des Managementsystems ein Web Frontend gewählt, da diese weitestgehend[1] plattform- und programmiersprachenunabhängig sind. Zusätzlich soll die Oberfläche den Nutzer auf logische Weise durch die Konfigurationsebenen führen und ihm nützliche Optionen anbieten können.

- **Vielseitige Einsatzfähigkeit**
  Das Managementsystem soll eine universelle Form haben, das heißt, es soll weitestgehend unabhängig vom Zielsystem sein und nur durch geringfügige Änderungen angepasst werden.

Die letzten beiden Kriterien richten sich weniger an die anzuwendende Technologie, sondern eher an die Gestaltung des Managementsystems und damit an den Entwickler. Ob die aufgeführten Kriterien umsetzbar sind, und welche Technologie dafür am besten geeignet ist, wird in den folgenden Abschnitten genauer untersucht.

3.2 Apache Webserver und PHP

Die Kombination von Apache Webserver und PHP wird in der derzeitig eingesetzten Version des Managementsystems genutzt. Dabei benötigt der Apache Webserver mit dem PHP Modul eine Festplattenkapazität von ca. 5 Megabyte und konstant 12 Megabyte Hauptspeicher. Bei der CPU Lastmessung wurden Spitzenwerte von 15 Prozent gemessen. Allerdings beschränkt sich der Umfang nur auf die Basisfunktionalität. PHP eignet sich hervorragend für den Einsatz in kleinen Projekten mit relativ wenig Anwendungslogik. Die Ausführungsgeschwindigkeit ist stark abhängig von der Größe und Komplexität des eingesetzten Skriptes. Die Lizenz des Apache Web Servers erlaubt eine Veröffentlichung mit eigenen Auflagen. Die Zend Engine, die Laufzeitumgebung von PHP, wird auch als Open-Source-Software publiziert und besitzt eine Apache-ähnliche Lizenz. Damit können die mit PHP erstellten Anwendungen ohne Bedenken veröffentlicht werden.

3.3 Apache Tomcat mit Cocoon und XSP

Apache Tomcat ist ein Servlet Container, der als Referenzimplementation der *Java Servlet* und der *Java Server Pages* Technologie verwendet wird. Die Spezifikationen für diese Technologien werden dabei von Sun entwickelt. Er basiert vollständig auf Java und wird durch XML Dokumente konfiguriert. Seine Hauptaufgabe ist das Ausführen von Servlets, die kleinen Programme, die serverseitig ausgeführt werden können. Zusätzlich beinhaltet Tomcat den JSP-Compiler *Jasper*, der die Java Server Pages in Servlets umwandelt, kompiliert und anschließend ausführt. Der Apache Tomcat eignet sich hervorragend für die Erzeugung dynamischer Inhalte aus großen javabasierten Softwareprojekten. Der integrierte HTTP-Server dient der Ausgabe von statischen Inhalten, ist aber viel langsamer als der

[1] Auch zwischen den Browsern existieren Unterschiede bei der Darstellung von Webseiten. Diese lassen sich aber durch geeignete Programmiertechniken mit in der Konzeption berücksichtigen.

traditionelle Apache Webserver. Deshalb wird in der Praxis der Webserver mit einem Plugin eingesetzt, welches die Anfragen für dynamische Inhalte direkt an Tomcat weiterleitet und nur statische Inhalte direkt an den Client zurücksendet. Die Kommunikation zwischen Webserver und Tomcat Applikationsserver erfolgt dabei über das *Apache JServ Protocol*.

Das „Publishing Framework Servlet“ Cocoon ist ein javabasiertes Framework und nutzt die Servlettechnologie für die Ausgabe. Es hat die Aufgabe, auf Anfrage des Clients die in XML vorliegenden Quelldaten mit Hilfe der XSL Transformation in XHTML, PDF, RTF oder andere umzuformen. Das Ziel dieses Frameworks ist die getrennte Ablage der Inhaltsdaten in XML Dateien, damit sie vollständig unabhängig von Anwendungslogik und Layout sind. Das Layout wird in Stylesheet-Dateien, der dynamische Inhalt in einer XSP(*eXtensible Server Page*) abgelegt. Diese Trennung bewirkt eine einfache und effektive Weiterverwendung und verringert den Overhead für das Management. Mit zusätzlichen Erweiterungskomponenten wie die *Portal Engine* oder das Content Managementsystem *Apache Lenya* lassen sich umfangreiche dynamische Webanwendungen gestalten. Da das Cocoon Projekt ebenfalls wie Tomcat ein Apache Projekt ist, unterliegt es der Apache Software Lizenz. Es gelten also auch hier die selben Nutzungsbedingungen wie bei dem Apache Webserver.

Die Sprache Java wird beim Kompilieren in einen Bytecode übersetzt, der zu einem späteren Zeitpunkt vollständig reproduzierbar ist. Zum Sichern des Quellcodes besteht nur die Möglichkeit, den Bytecode auf der Zielmaschine in Maschinencode umzuwandeln. Das Wurzelverzeichnis des Apache Tomcat Servers beinhaltet ein Vielzahl eigener Bibliotheken und das Cocoon Framework. Zusätzlich bindet er zur Laufzeit Bibliotheken aus der Java Runtime Environment ein. Nachdem alle diese Dateien und Bibliotheken umgewandelt wurden, umfasste das gesamte Verzeichnis eine Größe von ca. 150 Megabyte. Die Ursache dafür liegt in dem Einbinden von fast allen Bibliotheken aus dem Java SDK, da diese Bibliotheken von Tomcat benötigt werden, um zur Laufzeit die Servlet Klassen zu erstellen. Bei der Ausführung eines Servlets wurden zusätzlich die folgenden Werte gemessen:

- Bedarf an Hauptspeicher: mindestens 1,5 MB, maximal 147 MB
- CPU Auslastung: bis 97%

Gegenüber dem Apache Webserver mit PHP liegen diese Werte um vieles höher. Da das Kriterium der Code-Sicherheit mit höchster Priorität in die Bewertung eingeht, hat es wenig Sinn, nach anderen, kleineren und freien Implementationen der Java VM zu suchen. Die Ausführungsgeschwindigkeit ist durch das Kompilieren des Servlets bei der ersten Ausführung sehr gering, wird aber bei wiederholten Ausführungen nur neu kompiliert, wenn sich der Quelltext geändert hat. Dennoch kann man sagen, dass im Durchschnitt die Ausführungsgeschwindigkeit gering ist. Die Lizenzrechte für Sun's Binärdistributionen von Java Runtime Environment und SDK sehen vor, dass sie uneingeschränkt, aber ohne Modifikationen, genutzt werden dürfen. Also auch das mit einem SDK erstellte Java Programm darf zusammen mit der Runtime Environment verkauft werden, das SDK jedoch nicht. Für solche Zwecke bedarf es einer speziellen kommerziellen Lizenz von Sun. Damit will sich Sun die Reinheit der Java Grundschicht sichern. Jeder soll die gleiche Grundlage benutzen können, doch fast niemand, bis auf ausgewählte Firmen, darf das SDK verändern.

3.4 Zope und Python

Als Alternative zu den zwei vorhergehenden Technologien soll hier Python in Verbindung mit Zope behandelt werden. Zope ist ein Framework für die Erstellung von Webanwendungen. Ein Großteil des seit 1996 existierenden Projekts wurde in der Programmiersprache Python geschrieben, einige wenige leistungskritische Abschnitte in C. Zope leistet aber noch weitaus mehr, wie Erich Seifert[4] schreibt:

> *„Zope ist eine Plattform, auf der Site-Entwickler Anwendungen erstellen, die an Site-Designer und geschäftliche Nutzer weitergegeben wird, und Komponentenentwickler geben neue Produkte und Anwendungen an Zope-Benutzer weltweit weiter.“*

Die Stärken von Zope liegen dabei hauptsächlich in der Darstellung dynamischer Inhalte, sicherer Web Sites, in der Bereitstellung integrierter Netzwerkdienste und Skalierbarkeit. Es unterstützt die derzeit gängigsten, offenen Standards und ermöglicht damit die Vereinigung unterschiedlicher Daten. Die Veröffentlichung unter der Open-Source-Lizenz sorgt dafür, dass Zope frei verwendet werden kann und ständig weiterentwickelt wird.

Grundsätzlich ist der effektive Einsatz von Zope immer noch abhängig von den Eigenschaften der Programmiersprache Python. Die objektorientierte Skriptsprache Python arbeitet wie Java. Der Quellcode wird immer in eine Bytecode-Repräsentation überführt, welche dann von der Python Virtual Machine(VM) ausgeführt wird. Um unnötige Arbeitsvorgänge zu vermeiden, wird der erzeugte Bytecode in .pyc-Dateien abgelegt und nur bei Änderungen in der Quellcode-Datei neu erzeugt. Prinzipiell gibt es keine Unterschiede in der Ausführungsgeschwindigkeit zwischen dem Bytecode aus einer .pyc-Datei und dem Bytecode, der aus einer .py-Datei erzeugt wurde. Allerdings ist das Laden des Bytecodes aus der .pyc-Datei um ein Vielfaches schneller als das Parsen und Übersetzen einer .py-Datei. Die Ausnahme bilden die Main-Skripts. Diese werden immer zur Laufzeit verarbeitet, denn diese Skripts sind in der Regel klein und ein vorheriges Kompilieren würde keinen großen Performancegewinn erwirken.

Ähnlich wie Java nutzen die Python Objekte Bibliotheken aus der Python VM. Deshalb ist es sehr schwierig und ineffizient eine solche Anwendung in Maschinencode umzusetzen. Eine andere Möglichkeit der Sicherung des Quellcodes gibt es allerdings nicht. Die Ausführungsgeschwindigkeit von Python Skripten ist vergleichbar hoch mit der von PHP.

3.5 Java Client/Server-Anwendung

Eine reine Client/Server-Anwendung auf Basis von Java kommuniziert immer über TCP/IP Sockets. Der Socket ist eine Kombination aus der IP Adresse des Computers und einer Portnummer. Bei einer 32 Bit langen IP Adresse der Version vier hat der Socket die Form A.B.C.D:XXX, wobei A-D die Werte 0 bis 255 annehmen können und XXX eine beliebige Portnummer sein kann. Der Port ist eine logische Nummer, über die ein Netzwerkdienst mit anderen Computern kommunizieren kann. Das bedeutet, eine Server-Anwendung ist über einen fest definierten Port für alle anderen im Netzwerk befindlichen Computer ansprechbar. Nach dem Verbindungsaufbau zwischen zwei Sockets wird der Datenaustausch

durch Streams realisiert, wofür in den Bibliotheken der Java Virtuel Machine die notwendigen Klassen zur Verfügung stehen. Der Aufbau einer solchen verteilten Anwendung auf niedrigster Ebene hat den Vorteil, dass die Architektur an das individuelle Ziel der Anwendung angepasst und somit sparsam mit den Ressourcen umgegangen werden kann. Die Anwendung erlaubt eine asynchrone Kommunikation zwischen Server und Client. Es hat aber auch den Nachteil, dass der Implementierungsaufwand viel höher ist als bei den webbasierten Ansätzen, da hier zusätzlich noch die gesamte Netzwerkkommunikation umgesetzt werden muss.

Speziell für das zu konzipierende Managementsystem ist dieser Lösungsansatz ungünstig. Er erzwingt eine sowohl serverseitige, als auch clientseitige Installation der Software. Das erhöht auch den Aufwand bei eventuell auftretenden Wartungs- und Aktualisierungsarbeiten. Außerdem ist der Administrator damit nicht unabhängig von einem Computer, sondern muss die Clientanwendung auf jedem Computer, mit dem er arbeiten möchte, zusätzlich installieren. Das Hauptproblem dieser Lösung ist, wie auch bei Apache Tomcat in Verbindung mit Java, dass der Quellcode nicht effektiv vor der Wiederherstellung geschützt werden kann. Obwohl für diese Anwendung nur die Java Virtual Machine für die Ausführung benötigt wird, würde ein Umsetzen in Maschinencode zu viel Speicher benötigen. Die Ausführungsgeschwindigkeit ist ebenfalls, wie bereits in Kapitel 3.3 beschrieben, als gering einzustufen. Da der Entwurf einer Client/Server-Anwendung für die Testzwecke sehr umfangreich ist, können über den Ressourcenverbrauch keine genauen Aussagen gemacht werden. Der Bedarf an Hauptspeicher und Festspeicher ist stark abhängig von der Größe der verteilten Anwendung. Da aber auch hier Java zum Einsatz kommt, können die selben Eigenschaften der Java Virtual Machine bezüglich der Ausführung von Quellcode angenommen werden wie in Kapitel 3.3. Die Lizenzbedingungen von Sun gestatten es, die fertige Anwendung mit der Java Laufzeitumgebung(JRE[2]) auszuliefern. [5]

3.6 Vergleich und Schlussfolgerung

Die vorangegangenen Abschnitte haben die betrachteten Technologien näher beschrieben. Durch diese Analyse konnte sich die beste von ihnen herauskristallisieren und für den Einsatz im Managementsystem ausgewählt werden. Tabelle 3.1 trägt die gesammelten Fakten noch einmal zusammen und unterstützt die Entscheidungsfindung.

Das wichtigste Entscheidungskriterium war die Quellcode-Sicherheit. Die einzige Programmiersprache für die eine akzeptable Möglichkeit der Sicherung existierte, war PHP. Bei den Sprachen Java und Python kann die Sicherheit nur durch eine Transformation in Maschinencode erlangt werden, was allerdings mit einem steigenden Ressourcenbedarf abgewertet wird. Die dabei erhoffte Erhöhung der Ausführungsgeschwindigkeit war nur unwesentlich zu verzeichnen. Auch in Sachen Implementierungsaufwand hat PHP in Verbindung mit dem Apache Webserver am besten abgeschnitten. Die Programmiersprache Python in Verbindung mit dem Zope Framework hat zwar einen höheren Implementierungsaufwand, kann sich aber bei den Kriterien Ressourcenbedarf, Ausführungsgeschwindigkeit und Lizenzbedingungen mit PHP messen. Letztendlich sind beide Java-basierten Lösungen aufgrund der Programmiersprache ungeeignet. Das Cocoon Framework zeigte seine

[2] JRE ist die Abkürzung für Java Runtime Environment

Technologie / **Kriterium**	**Apache Webserver + PHP**	**Tomcat + Cocoon + Java**	**Zope + Python**	**Java Client/-Server**
Quellcode-Sicherheit	möglich	schwierig	schwierig	schwierig
Lizenzbedingungen	geeignet	ungeeignet	geeignet	geeignet
Ressourcenbedarf	gering	hoch	gering	hoch
Ausführungsgeschwindigkeit	hoch	gering	hoch	gering
Implementationsaufwand	gering	mittel	mittel	hoch
Gesamtbewertung	geeignet	ungeeignet	ungeeignet	ungeeignet

Tabelle 3.1: Die Entscheidungskriterien im Überblick

Vorzüge in der spezialisierten XML-Verarbeitung und der großen Entwicklergemeinde, die eine Weiterentwicklung des Projektes unterstützt. Die Client/Server-Anwendung hat am schlechtesten von allen abgeschnitten. Bei ihr waren keine dieser Vorteile vorzufinden. Deshalb ist sie absolut ungeeignet für den Einsatz im Xiranet Storage-Managementsystem. Aufgrund der hier erlangten Kenntnisse soll der Apache Webserver mit PHP zum Einsatz kommen.

4 Web 2.0 Technologien

Web 2.0 ist die Bezeichnung für eine Vielzahl von neuen Internettechnologien, die sich derzeit etabliert haben. Geprägt wurde der Begriff *Web 2.0* im Oktober 2004, als Dale Dougherty vom O'Reilly Verlag und Graig Cline von MediaLive die gleichnamige Konferenz vorbereiteten. Es existiert keine genaue Vorschrift, die klar definiert, welche Technologie zum Internet der zweiten Generation gezählt werden kann. Allerdings hat Dougherty in [6] viele Schlüsselprinzipien für Anwendungen herausgearbeitet, die im folgenden aufgelistet werden:

- Das Web dient als Plattform für Anwendungen, die die Grenzen des lokalen Rechners überschreiten. Durch die Möglichkeit der Vernetzung von Anwendungen wird die strikte Trennung von lokaler und zentraler Datenhaltung aufgehoben. Die Nutzer können ihre Datenmengen auf zentralen Servern ablegen und die verteilte Anwendung kann lokale Daten einbinden. Teile der Anwendung selbst werden ausgelagert, Module werden bei Bedarf nachgeladen und Software aktualisiert sich von selbst. Sogar ganze Geschäftsprozesse können auf entfernte Server verlagert werden, um Ressourcen zu schonen.

- Die Anwendungen sind datengetrieben. Das bedeutet, dass der Aspekt der Wiederverwendung von Inhalten wichtiger ist als die Darstellung von Inhalten. Aus dieser verteilten Nutzung von Inhalten und Diensten lassen sich einfache Geschäftsmodelle ableiten. Mit der technischen Erweiterung der Möglichkeiten hat sich auch die Rolle des Anwenders verändert. Die Nutzer haben zunehmend die Aufgaben der Autoren übernommen und verwalten somit selbst die Informationen im Internet. Bei einem Weblog[1] zum Beispiel verlagert der Nutzer private Informationen in den Bereich der Öffentlichkeit.

- Die Dienste lassen sich mit Hilfe von offenen Schnittstellen und der Verwendung von Komponenten problemlos verknüpfen. Mehrere Dienste können dadurch zu einem neuen Dienst komponiert werden und der Anwender selbst benötigt kein Wissen darüber, wo die Informationen im einzelnen gewonnen werden. Eine entscheidende Rolle darin spielt die Web Services Technologie, die häufig in Internetportalen zum Einsatz kommt.

Doch nach welchen Maßstäben werden Anwendungen der Web 2.0-Kategorie zugeordnet? Mit dieser Frage hat sich auch Tim O'Reilly in [6] beschäftigt und in einem Brainstorming eine Gegenüberstellung von Web 1.0 und Web 2.0 verfasst, die in Auszügen in Tabelle 4.1 gezeigt wird.

[1] Weblog ist eine Wortkreuzung von Web und Log, manchmal auch nur Blog genannt. Es bezeichnet ein Tagebuch im Internet, in dem die letzten Einträge an oberster Stelle stehen

Web 1.0		**Web 2.0**
DoubleClick	➠	Google AdSense
Ofoto	➠	Flickr
Akamai	➠	BitTorrent
mp3.com	➠	Napster
Britannica Online	➠	Wikipedia
personal websites	➠	blogging
domain name speculation	➠	search engine optimization
page views	➠	cost per click
screen scraping	➠	web services
publishing	➠	participation
content management systems	➠	wikis
directories (taxonomy)	➠	tagging ("folksonomy")
stickiness	➠	syndication

Tabelle 4.1: Die Bedeutung von Web 2.0 (nach [6])

Erklärung zu Tabelle 4.1 Die Firma **DoubleClick** ist ein Anbieter von adaptiven Werbebannern. Dabei sammelt DoubleClick gezielt Informationen über den Besucher, zum Beispiel welche Seiten er bisher besucht hat und was für Inhalte ihn interessieren. Aus diesen Informationen wird dann das Werbebanner individuell für den Besucher einer Site zugeschnitten. Der Benutzer wird dabei über Cookies[2] eindeutig identifiziert. **Google AdSense** ist eine Weiterentwicklung dieser Grundidee. Ein Website-Betreiber kann die AdSense Werbefelder an beliebigen Stellen in der Website einsetzen und damit Geld verdienen. Der Inhalt der Werbefelder wird automatisch von Google erzeugt und ist abhängig von dem Inhalt der Website. Im Gegensatz zu DoubleClick werden keine Interessen des Benutzers gespeichert. Google analysiert nur die Inhalte der Website und speichert diese auf den eigenen Servern ab.
Das Unternehmen **Ofoto** bietet angemeldeten Nutzern die Möglichkeit, ihre Digitalfotos auf die Website hochzuladen, zu bearbeiten und für andere berechtigte Nutzer zugänglich zu machen. Zusätzlich können alle Mitglieder die Fotos bestellen und frei Haus ausliefern lassen[3]. Die Webanwendung **flickr** bietet den Mitgliedern ebenfalls die Möglichkeit der Veröffentlichung von Digitalbildern an. Die Weiterentwicklung besteht in der Einteilung in Kategorien, die auch Tags genannt werden, die Suche nach Stichworten, die Veröffentlichung mit Photostreams oder RSS-Feeds[4], die Kommentierung der Bilder und die Veröffentlichung unter frei wählbaren Lizenzen.
Akamai Technologies ist ein weltweit führender online Dienstanbieter, dessen Erfolg auf dem Prinzip der Lastverteilung beruht. Mit über 15.000 Servern in 69 Ländern werden renommierte Firmen wie eBay und Microsoft vertreten. Das Filesharing-System **BitTorrent** vertritt ebenfalls die Interessen der Lastverteilung. Die Peer-to-Peer-Verbindung wird

[2] Cookies sind Informationseinheiten, die vom Browser meist in Form von Dateien abgelegt werden. Dadurch ist es möglich die Informationen über längere Zeit auf Clientseite zu speichern.
[3] Mehr Informationen dazu unter http://www.ofoto.eu.com
[4] RSS steht für *Really Simple Syndication* und ist ein XML-basierter Standard für kleine Informationsblöcke. Diese werden für die Vermittlung von kurzen Nachrichten verwendet.

allerdings nicht nur zwischen Server und Client ausgeführt, sondern auch auf alle anderen beteiligten Clients ausgelagert. Das bedingt eine Segmentierung der auszutauschenden Datei und der Server informiert alle Clients über den Bezugsort der jeweiligen Segmente. Auch hier wird wieder eine Verlagerung auf den Nutzer sichtbar. Während Akamai mehr Server aufstellen muss, um die Dienstqualität zu verbessern, müssen bei BitTorrent mehr Teilnehmer im System sein um selbiges zu erreichen.
mp3.com und **Napster** waren Musiktauschbörsen, die zu Beginn Musikstücke kostenlos zum Download bereitstellten. Das revolutionäre an Napster war dabei, dass es nur die Metainformationen zu den Musikstücken und ihren Anbietern gespeichert hat. Bei einer Suchanfrage an den Server sendete dieser die Information zurück, von welchem Client über Peer-to-Peer das gesuchte Musikstück bezogen werden kann.
Britannica Online ist eine amerikanische Online Enzyklopädie, die sich selbst als weltweit sicherste Informationsquelle bezeichnet[7]. Ihr seriöses Auftreten und ihr Erfolg hat den Ursprung in den traditionellen Medien und der Wissensaufbereitung seit 1768. **Wikipedia** dagegen ist eine jüngere Online Enzyklopädie und beschränkt sich auf den Informationsaustausch über das Web. Das besondere daran ist, dass jeder am Autorenprozess teilhaben kann.
Was früher die **personal websites**, die persönlichen Homepages waren, sind heute die Blogs geworden. Jeder Blog-Autor, der seine Erfahrungen der Öffentlichkeit zur Verfügung stellt, bekommt ein Feedback, was in der Community zu einer vermehrten Wissensverbreitung führt.
Während früher die Sites durch eindeutige Domainnamen an ihre Bedeutung gebunden wurden, werden sie jetzt durch Suchmaschinenoptimierung bekannt gemacht. Anfangs wurde die Werbung noch pro Seitenbesuch bewertet, jetzt wird sie pro Klick auf einen Link bewertet. Und auch bei der Informationsgewinnung wurden Fortschritte gemacht. Wenn man früher noch die HTML Seite der Google Suchergebnisse aufwendig parsen musste, ist es jetzt möglich die Web Services Schnittstelle des Dienstes zu nutzen und die Suchanfrage individuell durchzuführen. Der Übergang von der **Taxonomie** zur **Folksonomie** [5] bezieht sich auf die Klassifizierung von Inhalten, wobei man von der klassischen Gruppierung in Ordnern absieht und Tags für eine detailliertere Klassifizierung verwendet.

In dem Artikel stellt O'Reilly auch fest, dass der Begriff *Web 2.0* zu einem Modewort vieler Firmen geworden ist. Diese verwenden den Begriff oft, ohne eine genaue Bedeutung des Wortes zu kennen. Die wichtigsten Anwendungstechnologien, die zur zweiten Generation zählen, werden in den Folgenden Abschnitten erläutert.

4.1 Web Services APIs

Die Web Services Technologie spezifiziert Schnittstellen für die Kommunikation zwischen verteilten Anwendungen, die so genannten Web Services APIs. Durch diesen XML-basierten Datenaustausch sind die Anwendungen in der Lage, Objekte über die Systemgrenzen hinweg zu erzeugen und miteinander kommunizieren zu lassen. Die drei wichtigsten APIs sind

[5] Folksonomie ist die unkontrollierte Klassifizierung von Inhalten ohne Berichtigung durch eine höhere Instanz. Die Informationen werden also von den Personen kategorisiert, die sie auch benötigen

SOAP[6], UDDI[7] und WSDL [8]. Seit der Einführung im Jahr 1998 wurden dem Web Services Standard sehr viele Subspezifikationen hinzugefügt, die sich mit den gegenwärtigen Problemen wie beispielsweise Sicherheit und Transaktionen beschäftigen. [8]

4.2 AJAX

Der Begriff Ajax ist die Abkürzung für *Asynchronous JavaScript and XML*. Dabei ist es keine Technologie im engeren Sinne, sondern ein Ansatz für die Kombination der vorhandenen Technologien HTML/XHTML, CSS, JavaScript, Document Object Model, XML, XSLT und das XMLHTTPRequest-Objekt. Dieses Konzept ermöglicht die asynchrone Datenübertragung zwischen einem Browser und einem Server über das HTTP-Protokoll. Es können also schrittweise Updates einer Webseite durchgeführt werden, ohne die HTML-Seite neu laden zu müssen.

Für den Einsatz dieses so genannten Ajax-Modells muss der Browser die oben aufgezählten Technologien unterstützen und die folgenden Aufgaben bewältigen:

- Das Document Object Model(DOM) muss interpretiert werden können, damit Objektbäume dargestellt und manipuliert werden können.
- Das XMLHTTPRequest-Objekt muss vom Browser unterstützt werden, damit XML Dokumente asynchron vom Server angefordert werden können.
- Für die standardmäßige Darstellung von Inhalten wird HTML und CSS genutzt.
- JavaScript muss aktiviert sein, damit dynamische Inhalte erzeugt und das DOM zur Laufzeit genutzt werden kann.

Für die Übertragung der asynchronen Daten können REST[9] -ähnliche Verfahren, JSON[10], XML, SOAP und HTML verwendet werden.

Das Modell der klassischen Webanwendungen umfasst eine zustandslose Kommunikation, die immer durch einen clientseitigen Aufruf begonnen werden muss. Die Anfrage wird mit dem HTTP-Protokoll an den Server gesendet, welcher eine beliebige Anzahl von Abarbeitungsprozessen ausführt und anschließend ein Ausgabedokument im HTML-Format an den Client zurücksendet. Dieses klassische Modell beschreibt im allgemeinen die Nutzung von Hypertextmedien, ist allerdings unpassend für die Realisierung von Software Anwendung.

Das Modell der AJAX Webanwendungen nutzt eine Zwischenschicht auf der Clientseite, die AJAX-Engine. Die Engine übernimmt zwei Aufgaben, die Darstellung im Browser und die Kommunikation mit dem Server. Der Nutzer arbeitet mit der Anwendung unabhängig

[6] SOAP ist die Abkürzung für *Simple Object Access Protocol*, welches auf XML basiert und das Rahmenwerk für die Nachrichten spezifiziert

[7] UDDI steht für *Universal Discovery Description and Integration*, ein Dienst für das Auffinden von Web Services im Internet

[8] WSDL ist die *Web Service Description Language*, eine Schnittstellenbeschreibungssprache, die auf XML basiert und jeden Service eindeutig beschreibt

[9] REST steht für *restructured Text* und ist ein sparsames Übertragungsformat für Nutzdaten, in [9] näher beschrieben.

[10] JSON ist ein weiteres Datenaustauschformat, mehr dazu in [10]

von der Server-Kommunikation. Die benötigten Inhalte werden von der Engine asynchron nachgeladen und die Wartezeiten des Nutzers werden dadurch reduziert.

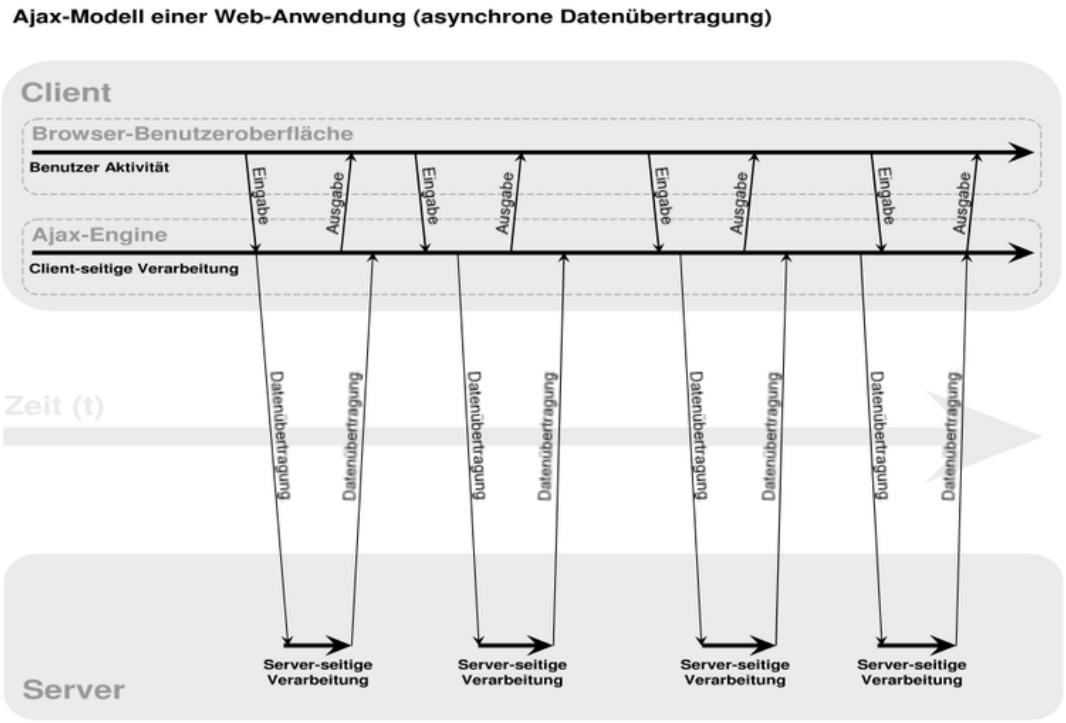

Abbildung 4.1: Das Ablaufdiagramm der Kommunikation zwischen Client und Server (Quelle: Daniel S. Haischt, www.adaptivepath.com)

Bei dem ersten Seitenaufruf wird eine initiale Seite an den Client gesendet. Dabei wird auch die Ajax-Engine, die eine JavaScript API ist, vom Server an den Client übertragen und initialisiert. Der weitere Verlauf der Aktualisierung kann in Abhängigkeit von der Struktur der Implementierung mit zwei Methoden durchgeführt werden.

1. Direkte AJAX-Implementierung
   Der Server stellt eine weitere Schnittstelle zur Verfügung, die die Ajax-Engine nutzt, um ausschließlich XML Daten auszutauschen. Die Engine selbst generiert dann dynamisch und zur Laufzeit die neuen Inhalte und fügt sie in die angezeigte Seite ein. Diese Methode spart die Ressourcen des Servers und den Traffic ein.

2. Indirekte Ajax-Implementierung
   Hierbei werden ganze HTML-Fragmente serverseitig erzeugt und an den Client übertragen. Dieser Vorgang dauert zwar länger als mit einer direkten Ajax-Implementierung, ist aber auch einfacher zu implementieren.

Viele große Webdienste-Anbieter bedienen sich dieser Technik, um dem Nutzer die Wartezeit zu verkürzen und die Vorteile aus der klassischen Desktop-Anwendung zur Verfügung

zu stellen. [11]

4.3 Abonnement-Dienste

Zu den Abonnementendiensten gehören RSS, RDF und Atom. **RSS** ist die Abkürzung für „Really Simple Syndication“ und bedeutet „wirklich einfache Verbreitung“. Dieses Synonym beschreibt eine Technologie, die für die einfache Informationsübertragung ohne Formatierungsinformationen entwickelt wurde. Die Inhalte werden mit XML formatiert und können von jeder RSS-fähigen Clientanwendung abonniert werden. In der Praxis sind solche Informationsangebote auf jeder größeren Site zu finden, die auch eine Community mit Informationen versorgen muss.

RDF, das *Ressource Description Framework*, ist der Vorgänger von RSS. Es ist eine formale Sprache zur Bereitstellung von Metadaten im Internet und wurde vom W3C[11] entwickelt. Mit der Hilfe von RDF können die Eigenschaften von Inhalten im World Wide Web in einer durch Maschinen lesbaren Form beschrieben werden. Diese Beschreibung ist die Voraussetzung für die Funktionalität des semantischen Web, bei dem Anfragen durch ihre Bedeutungsinhalte bearbeitet werden. Hauptsächlich liegen diese Beschreibungen in RDF-Syntax als XML-formatierte Dateien vor, es besteht aber auch die Möglichkeit der Darstellung als RDF-Modell in Graphenform. [12]

Atom kann als Nachfolger von RSS gesehen werden. Es fasst die vielen Subspezifikationen des RSS Formates zu einem Format zusammen und erweitert die Ausrichtung an den Bedürfnissen von Nachrichtenseiten und Weblogs. Entwickelt wurde das *Atom Syndication Format* von einer Vereinigung namens *AtomEnabled Alliance*. Atom wurde im Dezember 2005 als RFC 4287[12] veröffentlicht und ist damit ein offizieller Internetstandard. [13]

4.4 Soziale Software

Soziale Software ist die allgemeine Bezeichnung für Software, die eine menschliche Kommunikation, Interaktion und Zusammenarbeit ermöglicht. Bekannte Beispiele für soziale Software sind Kontaktbörsen, Foren, Internet Relay Chats und eLearning-Plattformen. In Verbindung mit der zweiten Generation des Internets haben sich die Weblogs und Wiki-Systeme etabliert.

Die **Weblogs**, die auch kurz **Blogs** genannt werden, sind Online-Tagebücher mit Kommentarfunktion. Somit hat jeder autorisierte Nutzer die Möglichkeit einen Artikel zu verfassen, der von jedem Nutzer kommentiert werden kann. Die Kommentare am Ende des Artikels sind meistens Korrekturen oder Erweiterungen der Leser zu dem jeweiligen Thema. Die Inhalte der Blogs reichen dabei von Freizeitgestaltung bis hin zu fachlichen Themen.

Die **Wiki-Systeme**, oder auch kurz **Wikis** genannt, sind Online-Seitensammlungen für die gezielte Informationsverbreitung. Genauer betrachtet sind Wikis spezielle Content Ma-

[11] Das *World Wide Web Consortium* ist ein Standardisierungsgremium für Richtlinien von Internettechnologien. Es wird in Kurzform W3C genannt.

[12] Die RFC 4287 kann unter http://www.ietf.org/rfc/rfc4287.txt eingesehen werden.

nagementsysteme, in denen jeder Nutzer auch die Rolle des Autors übernehmen kann. Es kann auch als Werkzeug für das Wissensmanagement bezeichnet werden, da der Hauptschwerpunkt das Abspeichern und zur Verfügung stellen von Informationen ist. Der einfache Aufbau dieser Systeme bewirkt, dass eine breite Zielgruppe diese Systeme nutzt. Das wohl bekannteste Wiki ist Wikipedia, die Online-Enzyklopädie, aber auch Entwicklergruppen und andere Interessengruppen nutzen diese Systeme für die Informationsverbreitung. Sogar der Einsatz in firmeninternen Netzwerken hat zugenommen, da die Nutzung fast so einfach ist wie Emails schreiben und ein Wiki damit einfacher zu bedienen ist als große Enterprise Dokumentenmanagementsysteme. [14]

Diese Technologien bilden die Basis von *Web 2.0*, nach der Auffassung von Andrew Keen[15] die Basis einer revolutionären Bewegung. Mit Recht stellt Keen auch die sozialkritische Frage nach der Zukunft von Web 2.0 und ob diese Entwicklung wirklich nur Vorteile mit sich bringt. Wenn jeder Anwender den Vorteil nutzt ein Autor zu sein und ohne Einschränkungen seine Gedanken veröffentlicht, wer sollen das alles lesen, und wer kontrolliert die Inhalte?

Die Web2.0 Technologien wurden nicht in Kapitel 3 aufgeführt, da sie nicht mit den anderen vier Technologien vergleichbar sind. Speziell für den Einsatz am Storage-Managementsystem sind nur die AJAX-Technologien interessant. Diese werden auf jeden Fall zum Einsatz kommen, da eine asynchrone Datenkommunikation für die Ausgabe des Systemstatus benötigt wird. Diese Technologien agieren aber hauptsächlich auf der Clientseite, weshalb sie auch nicht als serverseitige Basistechnologien betrachtet werden können.

5 Analyse von Managementsystemen bestehender Storagelösungen

5.1 Anforderungen an ein Managementsystem

Das Managementsystem einer Storagelösung soll dem Benutzer die Möglichkeit geben, auf einfachste Art und Weise Änderungen an der Hardwarekonfiguration vorzunehmen. Die mit dem System zu erreichende Zielgruppe spaltet sich in zwei Kategorien. Zum Einen in Administratoren, die nur über ein Grundwissen auf diesem Gebiet verfügen und nicht viel Zeit für eine intensive Vertiefung dieses Wissens haben werden. Zum Anderen in Administratoren mit gutem oder sehr gutem Fachwissen, deren Ansprüche an die Konfigurationsmöglichkeiten um vieles höher liegen. Nun gilt es, einen Mittelweg für beide Zielgruppen zu finden. Der Administrator soll sicher durch die Konfigurationsroutinen geführt werden und trotzdem die Möglichkeit haben, eigenständig Einstellungen vorzunehmen. Die graphische Oberfläche der Webanwendung soll ihn dabei unterstützen und die schwierigen Aufgaben abnehmen. Sie muss folglich inhaltlich klar strukturiert sein und darf den Anwender nicht verwirren. Das Hauptziel für den Anwender ist die komfortable Konfiguration der Hardware.

In Anlehnung an die Arbeiten von [16] und [17] wurden Kriterien ausgearbeitet, welche für den Vergleich von Managementsystemen notwendig sind. Sie werden im Folgenden Abschnitt näher beschrieben.

5.2 Bewertungskriterien

- **Technische Kriterien**
  Diese Kriterien beschäftigen sich mit dem technischen Inhalt eines Managementsystems. Sie untersuchen Installationsart und -Aufwand, sowie den damit verbundenen Anpassungsaufwand. Weiterhin ist interessant, welche Technologie server- und clientseitig eingesetzt wurde, und ob eine asynchrone Kommunikation zwischen beiden möglich ist. Für den Administrator ist es außerdem interessant, ob das Managementsystem skalierbar ist und die im Cluster befindlichen Geräte verwalten kann.

- **Benutzerfreundlichkeit** (Usability)
  Die Analyse der Benutzerfreundlichkeit betrachtet die intuitive Bedienbarkeit des Frontends, ob das System mehrsprachig ist und ob Seitenstruktur bzw. die Navigationsstruktur und das Navigationskonzept für den Benutzer logisch und erkennbar sind. Für den Benutzer ist auch interessant, wie hoch der Einarbeitungsaufwand für ihn ist und in welchem Berechtigungskontext(Rollen, Gruppen etc.) er sich bewegt.

Existieren vielleicht unterstützende Hilfen bei der Bedienung, wie beispielsweise automatisches Vervollständigen oder Assistenten? Auch die Ladezeiten und Fortschrittsanzeigen bei lang anhaltenden Systemprozessen sind für den Benutzer wichtig, damit er immer den aktuellen Zustand des Systems im Blick hat. Nützlich für jeden Neuling auf dem Gebiet dieser Systeme sind Hilfefunktion in Hauptsprache und kontextsensitive Hilfen. Besser ist ein Handbuch oder eine ausführliche Online-Dokumentation.

- **Design** (Layout und Ästhetik)
  Das Design ist der Grundstein für die Benutzerfreundlichkeit. Ein gutes Design ist gekennzeichnet durch klare Seiten und übersichtliche Strukturierung. Das Seitenlayout ist stets gleich bleibend, und Schriftgröße und Schrifttyp werden so gewählt, dass Textinhalte gut lesbar sind. Zudem sollte eine sinnvolle Farbwahl getroffen werden, damit das Auge des Benutzers die wichtigen Elemente von den unwichtigen unterscheiden kann.

- **Inhalt** (Dienstqualität der Managementanwendung)
  Bei der inhaltlichen Analyse soll überprüft werden, ob die Systeme dem Benutzer ausreichende Konfigurationsmöglichkeiten für ein Verwaltungsobjekt bieten können. Zum Einen soll dem Benutzer so viel wie möglich Arbeit abgenommen werden, zum Anderen soll er aber so viel wie möglich Einstellungen selbst vornehmen können. Diese Optionen sind stark von der installierten Hardware abhängig. Des Weiteren ist die Kombination und Strukturierung der Inhalte ausschlaggebend dafür, wie gut sich der Benutzer im System zurechtfindet, und damit auch eine Grundlage für die Usability.

5.3 Bewertung

Die Bewertung erfolgt nach den im vorhergehenden Abschnitt erläuterten Bewertungskriterien. Damit die Systeme miteinander vergleichbar sind, werden die Bewertungsnoten *gut*, *ausreichend* und *schlecht* eingeführt.

5.3.1 Cisco SN5428

Der Cisco SN 5428 ist ein Multiprotokoll Storage Router. Er agiert als Koppelelement zwischen Fibre Channel und Gigabit Ethernet Netzwerken. Der integrierte 8-Port Fibre Channel Switch übernimmt die Aufgabe des SAN Switches und die Zugriffskontrolle. Auch das Abbilden von iSCSI Gerätenamen auf physische Speicheradressen im Netzwerk, das *Mapping*, gehört zu seiner Aufgabe. Für die Konfiguration wird ein Managementsystem zur Verfügung gestellt, welches schon vorinstalliert ist.

- **Technik** Beim Erststart nach der Installation des Routers muss die IP Adresse an der Konsole eingegeben werden. Danach ist das Managementsystem über das Webinterface mit dieser Adresse zu erreichen. Es ist eine einfach gehaltene Webanwendung, die auf einem PHP-fähigen Webserver ausgeführt wird. Die Darstellung beschränkt sich dabei auf einfache HTML Seiten ohne eine asynchrone Kommunikation.

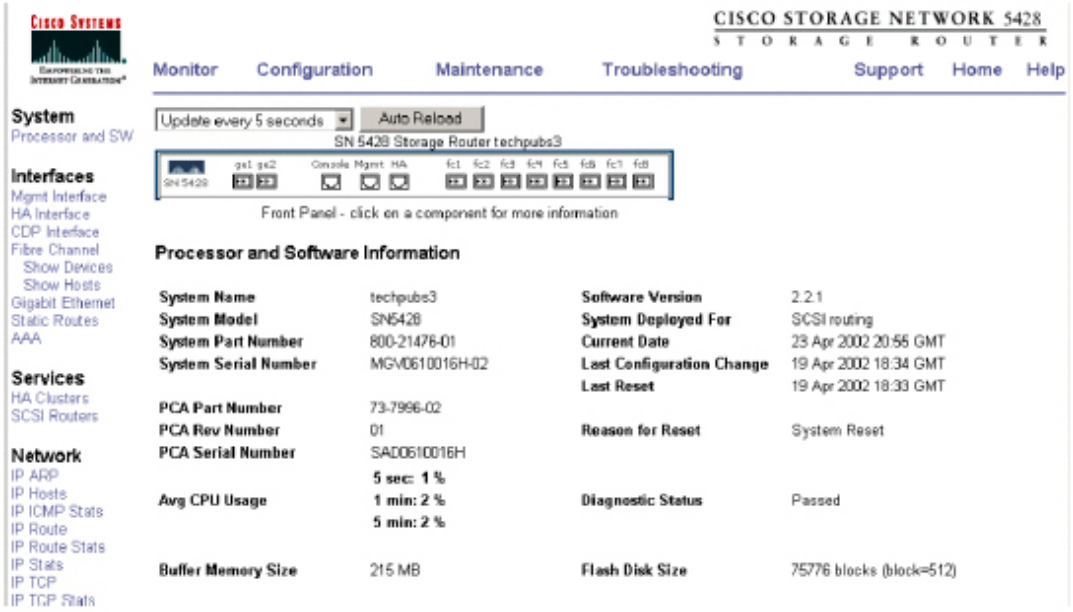

Abbildung 5.1: Die Managementoberfläche des SN5428 (Quelle: www.cisco.com)

- **Benutzerfreundlichkeit** Die klare Navigationsstruktur gibt eine Übersicht über den Aufbau der Seite. Der Benutzer hat die Möglichkeit sich als Admin oder Monitor anzumelden. Die schlicht gehaltenen Unterseiten erlauben eine intuitive Bedienbarkeit des Frontends und halten den Einarbeitungsaufwand gering. Eine kontextabhängige Hilfe existiert nicht, sondern nur ein Hilfebereich, der auch gleichzeitig das Onlinehandbuch ist. Der Benutzer muss folglich immer den Kontext wechseln, um die Bedeutung einer Option nachzulesen.
- **Design** Das Design ist schlicht und einfach, es wurden nur zwei Schriftfarben verwendet. Die Vielzahl der Navigationselemente lässt keine komfortable Navigation zu, so dass sie nur in Listenform vorliegt. Das Seitenlayout ist einheitlich gestaltet worden und Farbe sowie auch Schriftarten wurden sinnvoll ausgewählt. Durch den Einsatz von weiteren Layoutelementen sowie zusätzlicher Farben hätte der Aufbau der Seite besser strukturiert, und damit auch der Inhalt besser dargestellt werden können.
- **Inhalt** Der Inhalt der Site ist sehr umfangreich. Es existieren zahlreiche Einstellmöglichkeiten, die gruppiert nach ihrer Bedeutung dargestellt werden. Die trockene Darstellung der Inhalte zeigt, dass die Firma mehr Wert auf Funktionalität und inhaltliche Vollständigkeit legt als auf nutzerfreundliche Interaktionselemente.

Das System wurde mit der Note ausreichend bewertet.

5.3.2 Open-e iSCSI Enterprise

Die Firma Open-e bietet mit dem iSCSI Enterprise ein komplettes Software-System, mit dem die Hardware in einem SAN als *Device Server* agieren kann. Die Installation dieser Software ist denkbar einfach. Sie wird auf einem Flash RAM Modul ausgeliefert und muss nur in dem IDE Interface eingesetzt werden. Das Speichermodul enthält bei Lieferung ein

Open-e Betriebssystem und einen PHP-fähigen Webserver. Das im Betriebssystem eingebundene Managementsystem mit einem webbasierten Frontend wird nun näher untersucht.

Abbildung 5.2: Die GUI bei der Verwaltung von iSCSI Targets (Quelle: www.open-e.com)

- **Technik** Nach der Installation des Flash Moduls muss beim Starten des Betriebssystems nur die IP-Adresse in der Kommandozeile angegeben werden. Die restlichen Konfigurationen und zusätzlichen Einstellungen können über das Webinterface erledigt werden. Zu einem späteren Zeitpunkt kann der Administrator die wichtigsten Wartungsarbeiten über die Kommandozeilen-Schnittstelle durchführen. Serverseitig wird ein Webserver mit PHP eingesetzt, so dass der Administrator nur einen JavaScript-fähigen Browser für das Managementsystem benötigt. Für die Anzeige des Bearbeitungsstatus wird bei ausgewählten Operationen auch die AJAX Technologie eingesetzt.

- **Benutzerfreundlichkeit** Beim Einstieg in das Managementsystem kann der Benutzer sich eine Rolle aussuchen, in der er sich mit gültigem Passwort anmelden möchte.

Die danach folgende Benutzeroberfläche ist klar strukturiert nach Hauptkategorien und Unterkategorien. Jede Unterkategorie ist noch einmal visuell in Funktionsblöcke untergliedert, die über eine kontextbezogene Hilfe verfügen. Somit hält sich auch der Einarbeitungsaufwand des Benutzers in Grenzen. Die wichtigsten Einstellungen sind zwar vorkonfiguriert, ein Assistent oder Empfehlungen für spezielle Einstellungen existieren allerdings nicht. Die gesamte Präsenz läuft dabei in einem Browserfenster ab.

- **Design** Das Design ist schlicht und einfach gehalten. Ein zentriertes Inhaltsfenster lässt die Oberfläche in fast allen Bildschirmauflösungen erscheinen. Durch die kategorische Abgrenzung im Kopf der Seite setzt sich auch das untergeordnete Teilfenster, das den Inhalt repräsentiert, durch eine sehr gut angepasste Farbwahl ab. Die senkrechte Unternavigation auf der linken Seite vertieft noch einmal die Hierarchie. Es ist klar erkennbar, was Teilüberschrift und was veränderbares Element ist. Die Seiten sind nicht mit Inhalt überfüllt und geben klare Aussagen über das Wesentliche. Die Farben sowie Schrifttypen passen sich gut in das Gesamtbild ein.

- **Inhalt** Die Inhalte selbst werden klar und deutlich voneinander abgegrenzt, was auch zur Übersichtlichkeit beiträgt. Nachteilig an diesem System ist aber, dass ein Großteil der Hardwarekonfigurationen fest voreingestellt ist und nicht vom Benutzer verändert werden kann. Das fördert zwar die Übersichtlichkeit, schränkt allerdings auch die Flexibilität des Administrators ein.

Mit dieser Lösung wird dem Administrator viel Arbeit abgenommen, indem viele Einstellungen bereits festgelegt werden. Open-e legt damit mehr Wert auf das äußere Erscheinungsbild als auf den funktionalen Kern des Systems. Der Vorteil der Übersichtlichkeit wird mit dem Nachteil der fehlenden Flexibilität ausgeglichen. Deshalb wird dieses System mit *ausreichend* bewertet.

5.3.3 Xyratex StorView

StorView ist das Managementsystem der Firma Xyratex, welche versucht, die einfache Bedienung mit der Möglichkeit einer flexiblen Systemkonfiguration zu kombinieren. Das Ergebnis ist der StorView Global Manager, eine sichere webbasierte Verwaltungssoftware mit zahlreichen Funktionen.

- **Technik** Das Managementsystem wird von einen CGI-fähigen Webserver serverseitig ausgeführt. Auf der Clientseite werden die vielen dynamischen Interaktionen durch JavaScript und animierte GIFs unterstützt. Der technische Funktionsumfang ist bei diesem System am größten. Der Benutzer kann mit dieser Software global agieren, das heißt, er kann auch andere, entfernte Storagesysteme damit unter diesem Kontext administrieren.

- **Benutzerfreundlichkeit** Die gesamte Webanwendung ist am Modellbild des Storagesystems als Interaktionsmetapher ausgerichtet. Auf der linken Seite des Fensters befindet sich eine Navigation über alle zu verwaltenden Geräte in der Gruppe, auf der rechten Seite die Übersichtsseite zu dem ausgewählten Gerät. Der Benutzer bekommt auf der Übersichtsseite alle wichtigen Informationen. Er sieht sowohl den physischen

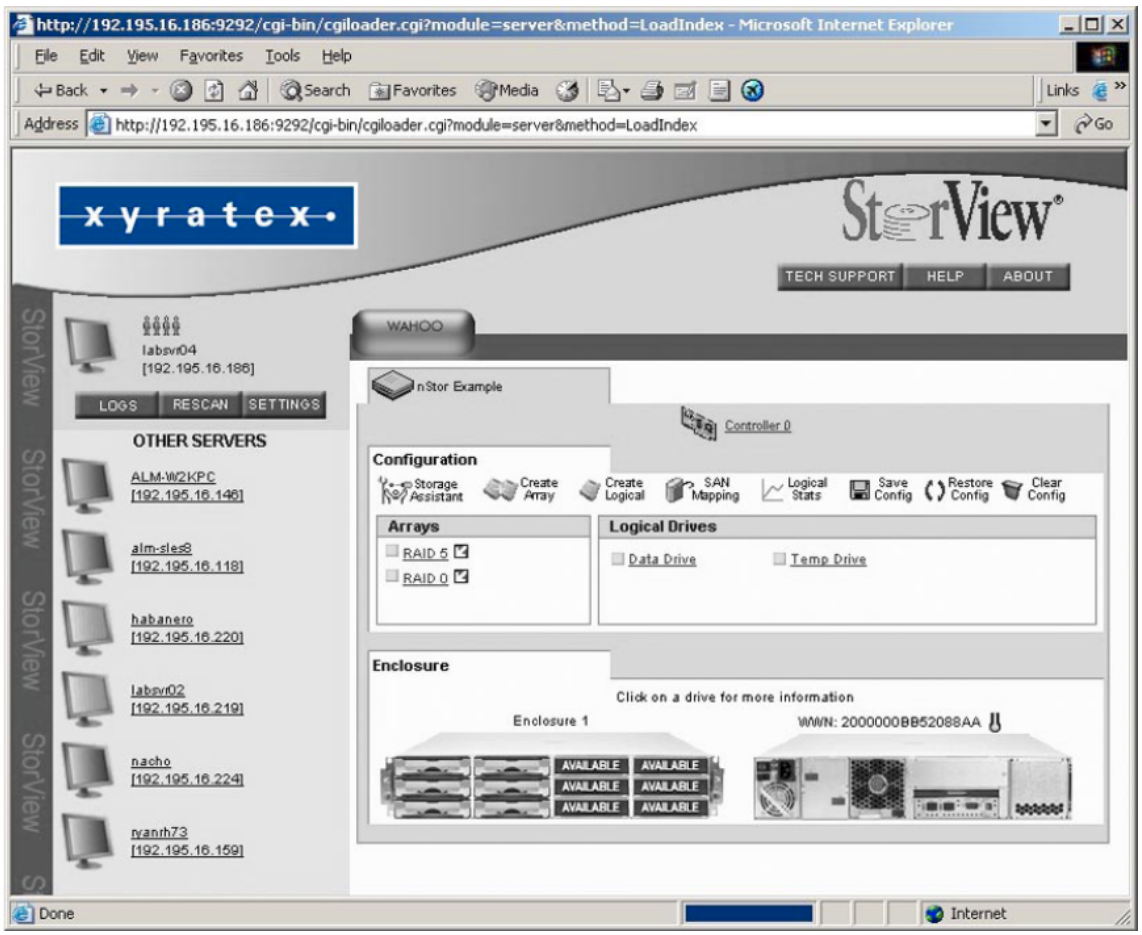

Abbildung 5.3: Die Startseite des StorView Managementsystems (Quelle: www.xyratex.com)

Status des Gerätes, als auch die logischen Konfigurationen des Systems. Die hierarchische Struktur mit einem gleich bleibenden Rahmenwerk führt den Nutzer gezielt zum gesuchten Menü. Der Vorteil der bildlichen Darstellung von Speicherhardware kann dann zum Nachteil werden, wenn eine Vielzahl an konfigurierbaren Speichermedien vorhanden ist, so dass die Übersichtlichkeit verloren geht und der Benutzer den Überblick verliert. Für eine leichtere Bedienung kann der Benutzer mit Drag-and-Drop-Operationen arbeiten. Die Anzeigeformulare sind gleichzeitig auch veränderbar, so dass der Bearbeitungskontext nur selten gewechselt werden muss. Zudem existiert ein Assistent, der den Benutzer bei der Erstellung von neuen RAID Arrays unterstützt und notwendige optimierte Voreinstellungen festlegt.

- **Design** Das Seitenlayout zieht sich konsequent durch die gesamte Webanwendung. Die Farben wurden passend zum Kontext gewählt und die Schriften sind gut lesbar. Die Struktur wurde in Abhängigkeit vom Inhalt logisch aufgebaut, allerdings vermindern die Vielzahl an Optionen die Übersichtlichkeit.

- **Inhalt** Die Inhalte wurden hierarchisch nach dem Aufbau der Hardware strukturiert. Bei den Hardwareeinstellungen kann der Benutzer im Vergleich zu den anderen

Lösungen sehr viel verändern.

Diese System besitzt nur Vorteile gegenüber den anderen Systemen und wird deshalb mit *gut* bewertet.

5.3.4 LSI Logic MyStorage Management Software

Das MyStorage Managementsystem von der Firma LSI Logic ist eine Client/Server--Anwendung, die für die Installation und Verwaltung von LSI Hardware in SAN-Netzwerken entwickelt wurde. Die Software zeigt Informationen über den Hostadapter, verfügt über Diagnosefunktionen und statische Auswertungsfunktionen, und bietet eine Konfiguration für die Multipath-Unterstützung, um mit einer Pfadredundanz die Ausfallsicherheit zu erhöhen.

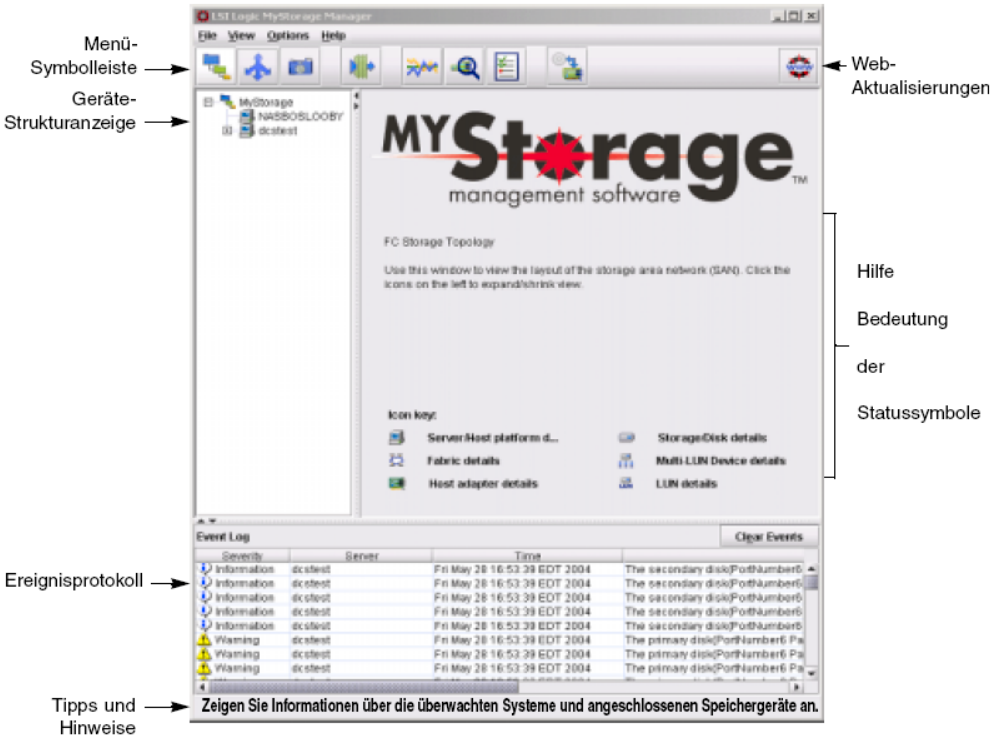

Abbildung 5.4: Das MyStorage Benutzerfenster mit Bezeichnungen (Quelle: www.lsilogic.com

- **Technik** Die Anwendung wurde server- und clientseitig mit der Programmiersprache Java implementiert. Aus diesem Grund muss sie auf beiden Seiten installiert werden. Jeder Hostcomputer, der die Serversoftware installiert hat, kann in der Clientsoftware als zu verwaltendes Objekt eingebunden werden. Die Kommunikation über Sockets erlaubt einen asynchronen Datenaustausch. Dadurch können Statusmeldungen und Statistiken in Echtzeit angezeigt werden. Der erhöhte Installationsaufwand bringt den Vorteil, dass die Clientanwendung nicht erst die Daten für die Darstellung herunterladen muss, sondern nur systemrelevante Daten über das Netzwerk übertragen werden. Das verringert die Ausführungszeit und Netzwerkbelastung.

- **Benutzerfreundlichkeit** Ziel der verteilten Anwendung ist es, die Installation und Konfiguration mit nur wenigen Mausklicks zu ermöglichen. Die klare Dreiteilung der Oberfläche und die Verwendung von eindeutigen Symbolen in den Menüs lässt den Benutzer intuitiv agieren. Das unten ausgerichtete horizontale Teilfenster zeigt die aktuellen Systemmeldungen als Link an, der mit einem Klick zum betroffenen Konfigurationsmenü beziehungsweise zum Problempunkt führt. Das erleichtert dem Benutzer die Arbeit, da er schneller zum Ziel gelangt. Die Geräte-Strukturanzeige im linken Teilfenster führt den Benutzer durch Hardwareeigenschaften des ausgewählten Hostcomputers. Durch die einheitliche Gestaltung der Anwendung wird der Einarbeitungsaufwand für den Benutzer im mittleren Bereich gehalten. Mit Assistenten, wie beispielsweise bei der Installation, und mit Fortschrittsanzeigen wird der Benutzer bei seinen Aktionen unterstützt. Eine kontextbezogene Hilfe existiert aber nicht, lediglich auf der Startseite der Anwendung befindet sich eine Erklärung der Menüleistensymbole. Ein ausführliches Handbuch in mehreren Sprachen kann von der LSI Logic Homepage bezogen werden.

- **Design** Das Design der Anwendung besteht aus einem typischen Java Layout, was in der letzten Version des Managementsystems durch weichere Formen verschönert wurde. Es ist übersichtlich strukturiert und Farbe sowie Schrift vermitteln einen seriösen Eindruck.

- **Inhalt** Die Inhalte werden nach der Rechnerhardware eines Hosts sortiert angezeigt. Der Benutzer hat dabei ausreichend Möglichkeiten um die Hardware zu konfigurieren.

In diesem System werden die Vorteile der verteilten Java-Anwendung genutzt. Wenige Nachteile finden sich allerdings in der Plattformunabhängigkeit und dem Design. Diese Bewertung kann mit einem *gut* abgeschlossen werden.

5.3.5 Intel® Storage System Software

Diese Software ist universell einsetzbar und kann mit jedem beliebigen Storagesystem von Intel® genutzt werden. Sie wird auch als *Storage System Console* bezeichnet. Ein einzelnes Storagesystem wird *Modul*(SSM) genannt.

- **Technik** Jedes Modul wird mit Betriebssystem und Managementsoftware ausgeliefert. Die serverseitige Software ist bereits vorinstalliert, clientseitig muss der Administrator die Console noch installieren. Es handelt sich um eine verteilte Java-Anwendung, die über Sockets kommuniziert. Der Installationsaufwand ist zwar höher als bei einer Webanwendung, der Vorteil liegt aber in der erhöhten Flexibilität der Anwendung. So ist auch eine asynchrone Kommunikation zwischen Server und Client möglich. Speziell dieses System ist skalierbar. Es können beliebig viele Intel® Storagesysteme in die Verwaltung einbezogen und als Gruppe verwaltet werden.

- **Benutzerfreundlichkeit** Der Umgang mit der Software wird durch die intuitive Bedienbarkeit der Anwendung stark erleichtert. Anders als bei den bisher untersuchten Webanwendungen ist der Einstiegspunkt der Software die oberste Abstraktionsebene der Hardware-Hierarchie. Im Hauptmenü bekommt der Benutzer alle eingebundenen Storagesysteme mit ihren Verbindungen untereinander angezeigt. Durch diese

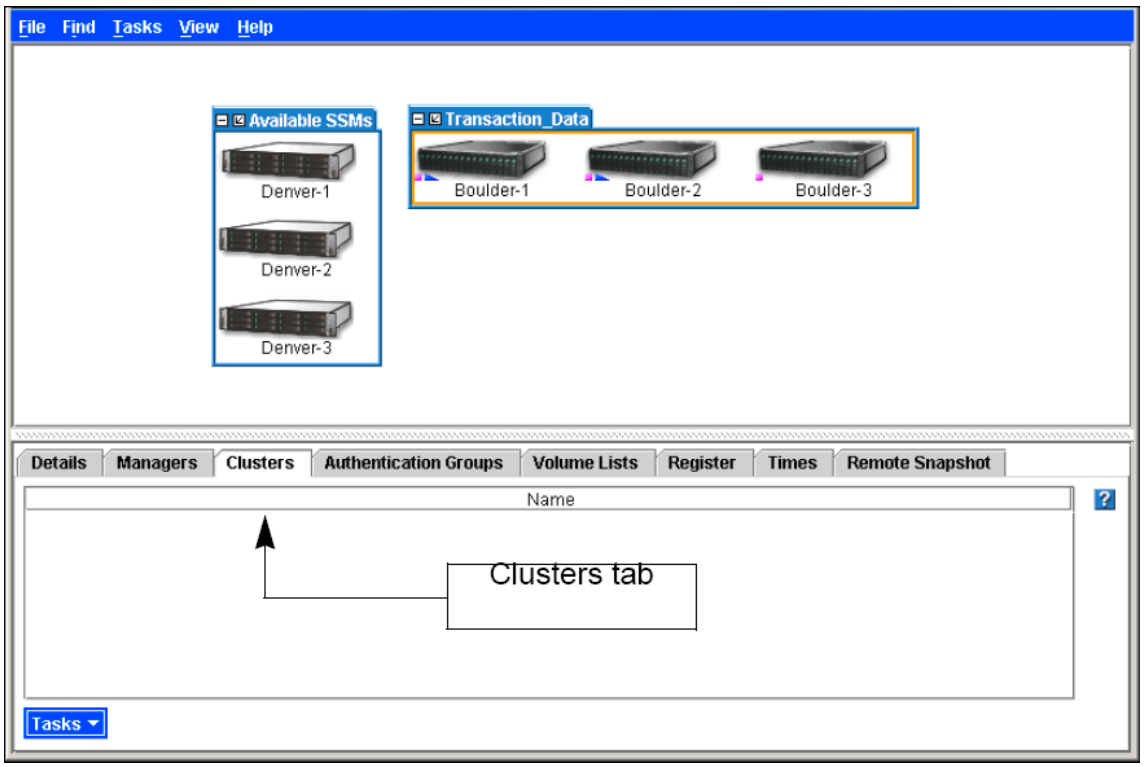

Abbildung 5.5: Die Oberfläche der Managementkonsole bei der Konfiguration eines Clusters (Quelle: www.intel.com)

Landkarten-Interaktionsmetapher bekommt er schnell den Überblick über beispielsweise die Vernetzung der einzelnen Module. Bei Auswahl eines Moduls springt die Software in eine niedrigere Konfigurationsebene, die nur das eine Modul verwaltet. Mit Hilfe von Assistenten für die schnelle Konfiguration wird der Einarbeitungsaufwand so gering wie möglich gehalten. Neben der Online-Hilfe, die über das Hilfe-Menü erreicht werden kann, existiert noch ein ausführliches Handbuch für die Software. Die Änderung von Moduleigenschaften kann für ganze Gruppen von Modulen durchgeführt werden.

- **Design** Die ausgewählte Interaktionsmetapher bewirkt, dass eine klare und übersichtliche Struktur der Anwendung vorliegt. Farbe und Schrifttyp wurden dezent ausgewählt und mit den zur Verfügung stehenden Mitteln der Java Plattform implementiert. Das Layout ist dabei gleich bleibend und die Seiten sind nicht mit Informationen überfüllt. Häufig werden für die Unterteilung in Unterkategorien Registerreiter verwendet, um die Übersichtlichkeit zu erhalten.

- **Inhalt** In dem Konfigurationsfenster eines Moduls werden sehr viele Einstellungen angezeigt, die auch verändert werden können. Die Konfigurationsebenen sind dort noch einmal in Funktionsklassen untergliedert.

Intel hat mit diesem System sehr viele Funktionalitäten verknüpft. Die Interaktionsmetapher der netzförmigen Anordnung von Teilsystemen ist in keinem anderen verglichenem

System zu finden und zeichnet sich als klarer Vorteil für große Systeme. Die einzige Verbesserungsmöglichkeit besteht in einer Überarbeitung der graphischen Oberfläche. Deshalb hier die Bewertung *gut*.

5.3.6 OpenFiler

OpenFiler ist eine CentOS Linux Distribution, die Open Source Software wie Apache, Samba, LVM2, ext3, iSCSI Enterprise Target und den Linux Kernel 2.6 enthält. Damit lässt sich aus herkömmlicher Hardware sowohl ein NAS-System als auch ein SAN aufbauen. Für die Konfiguration wird ein webbasiertes Management Interface eingesetzt, welches in den nächsten Punkten genauer untersucht wird.

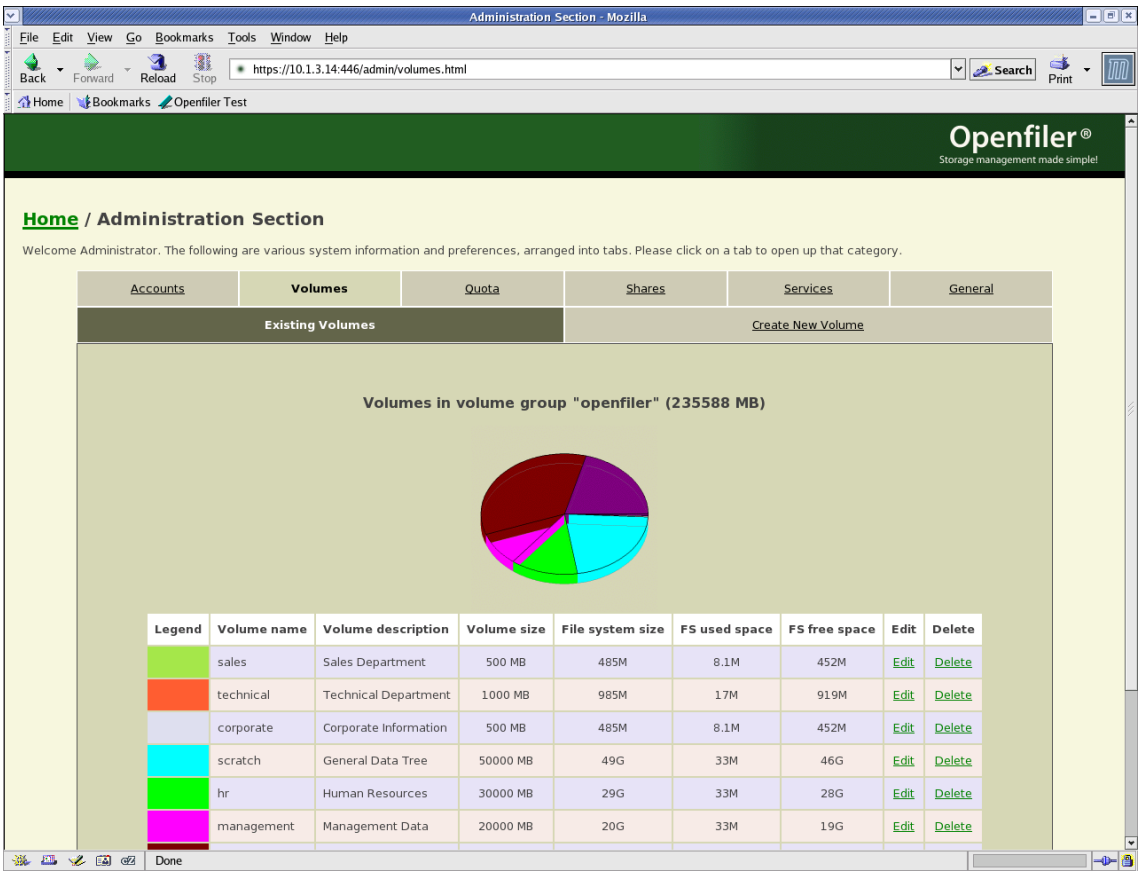
Administration Section - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
https://10.1.3.14:446/admin/volumes.html
Search Print
Home Bookmarks Openfiler Test
Openfiler®
Storage management made simple!
Home / Administration Section
Welcome Administrator. The following are various system information and preferences, arranged into tabs. Please click on a tab to open up that category.
Accounts Volumes Quota Shares Services General
Existing Volumes Create New Volume
Volumes in volume group "openfiler" (235588 MB)
Legend Volume name Volume description Volume size File system size FS used space FS free space Edit Delete
sales Sales Department 500 MB 485M 8.1M 452M Edit Delete
technical Technical Department 1000 MB 985M 17M 919M Edit Delete
corporate Corporate Information 500 MB 485M 8.1M 452M Edit Delete
scratch General Data Tree 50000 MB 49G 33M 46G Edit Delete
hr Human Resources 30000 MB 29G 33M 28G Edit Delete
management Management Data 20000 MB 20G 33M 19G Edit Delete
Done

Abbildung 5.6: Die Oberfläche des OpenFiler Webfrontend in der Konfigurationsseite für Volumen (Quelle: www.openfiler.com)

- **Technik** Der Installationsaufwand ist sehr gering, da das Managementsystem durch die Installation des Betriebssystems automatisch installiert wird. Serverseitig wird der PHP-fähige Apache Webserver eingesetzt, clientseitig kann jeder Browser mit HTML und JavaScript-Unterstützung genutzt werden. Die Kommunikation erfolgt dabei ausschließlich über das *Secure HTTP* Protokoll.
- **Benutzerfreundlichkeit** Die Gestaltung der Oberfläche ist sparsam und sehr einfach. Aus dem horizontal oben angeordneten Navigationsbalken kann keine genaue

Struktur des Inhaltes abgeleitet werden. Dementsprechend schwer fällt dem Benutzer die intuitive Bedienbarkeit des Frontends. Mit dem Rollenkonzept bekommt er den nötigen Zugriff auf Einstellungen und Statistiken. Allerdings existieren keine Assistenten und kontextsensitive Hilfen. Lediglich ein englischsprachiges HTML Dokument für Administratoren auf der Homepage des Projektes[1] unterstützt den Benutzer.

- **Design** Die Farbe ist schlicht und einfach, an manchen Stellen allerdings unpassend ausgewählt worden. So auch die Navigation, die aufgrund ihrer schlechten Farbwahl verwirrende Angaben über die aktuelle Menüposition gibt. Das Grundlayout ist auf jeder Seite gleich, und auch die Schriften wurden angepasst. Die Menge des Seiteninhaltes ist bei einigen Unterseiten zu hoch.

- **Inhalt** Inhaltlich wurden die Einstellungskategorien klar voneinander abgegrenzt und sinnvoll miteinander in Zusammenhang gebracht. Der Umfang an Funktionen ist relativ hoch. Auch bei diesem System wird deutlich, dass die Funktionalität der Anwendung wichtiger ist als die Gestaltung dieser.

Im Vergleich mit den Vorhergehenden Systemen konnten sehr viele Nachteile gefunden werden. Es wurde nur das nötigste an Funktionalitäten umgesetzt, um die Speicherhardware zu verwalten. Deshalb wurde dieses System mit *schlecht* bewertet.

5.3.7 FreeNAS

FreeNAS ist ein Open Source NAS Server, der auf FreeBSD 6 basiert und die notwendigen NAS Anwendungen zur Verfügung stellt. Er benötigt weniger als 16MB Speicher und kann auf Flash-Karten oder USB Sticks installiert werden. Zu den unterstützten Anwendungen gehören Samba, FTP, NFS, RSYNC Protokolle, lokale Benutzerauthentifizierung, Software RAID (Typ 0,1,5) und eine webbasierte Konfigurationsoberfläche für das gesamte System, welche in den folgenden Punkten genauer untersucht wird.

- **Technik** Nach der Installation von FreeNAS ist auch das Managementsystem betriebsbereit, eine weitere Konfiguration ist nicht notwendig. Es kann über die Standardports des HTTP- oder Secure HTTP-Protokolls aufgerufen werden. Serverseitig kommt ein PHP-fähiger Webserver zum Einsatz, clientseitig wird ein einfacher Browser benötigt. Eine asynchrone Kommunikation zwischen Client und Server wird allerdings nicht eingesetzt.

- **Benutzerfreundlichkeit** Aufgrund der Gestaltung der Oberfläche ist das Frontend intuitiv bedienbar. Der Seitenaufbau ist klar strukturiert, das Navigationskonzept ist einfach zu erkennen und der Benutzer bekommt schnell einen Überblick. Auch das Angebot von mehreren Sprachen der Oberfläche ermöglichen ihm das schnelle einarbeiten in das System. Für die Bedienung gibt es keine Hilfsfunktionen und keinen Hilfebereich. Bei einigen Konfigurationsmasken existiert ein kurzer Hinweistext. Auf der Homepage von FreeNAS steht außerdem ein Handbuch im PDF-Format für die Installation und Anwendung des Systems zum Download bereit. Auch in diesem System wird die Rollenmetapher für die Benutzeridentifizierung genutzt. Fortschrittsanzeigen für lang anhaltende Abarbeitungsprozesse kommen in diesem System nicht zum Einsatz.

[1] http://www.openfiler.com/docs/manual/

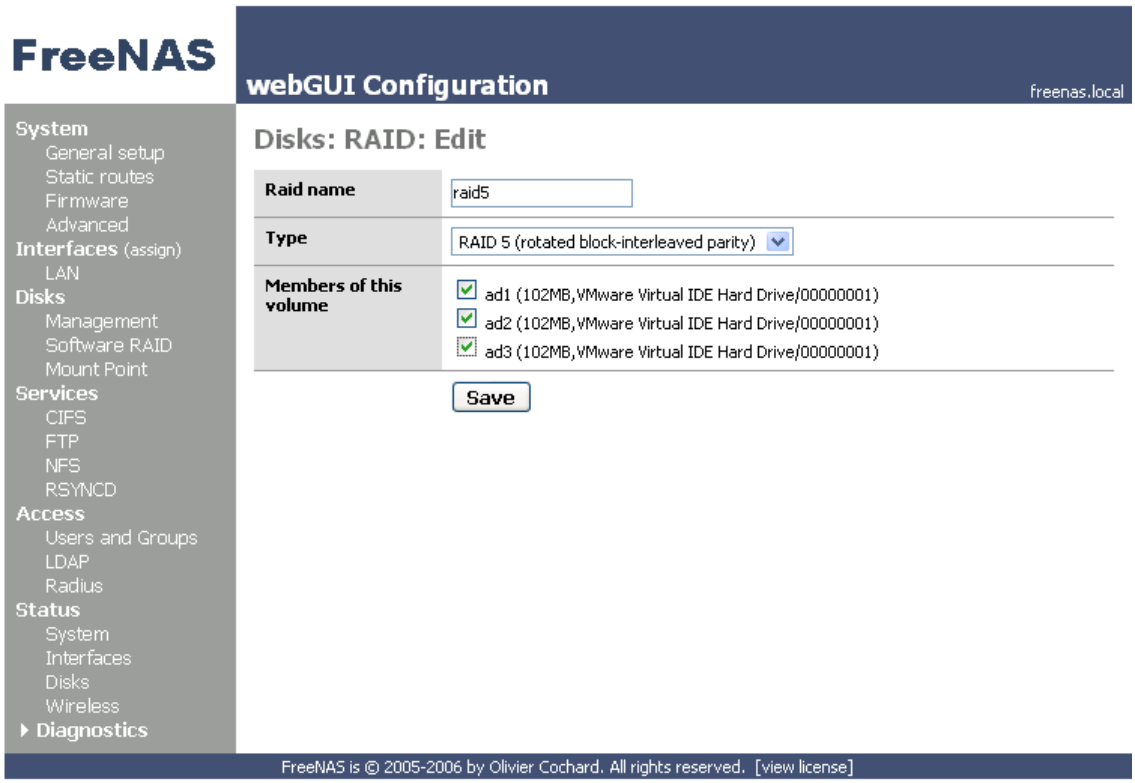

Abbildung 5.7: Die FreeNAS Konfigurationsoberfläche bei der Veränderung der RAID Einstellungen (Quelle: www.freenas.org)

- **Design** Das Design der Oberfläche wurde sehr gut gestaltet. Es ist klar und übersichtlich strukturiert, die Farben wurden passend ausgewählt und die Schrift in Typ und Größe angepasst. Das Seitenlayout wurde konsequent auf jeder Seite beibehalten. Die Menge der Informationen jeder Unterseite sind angemessen ausgewählt worden.

- **Inhalt** Inhaltlich wurden die Einstellungsmöglichkeiten auf die nötigsten Elemente beschränkt. Die Struktur des Inhaltes wurde nach Verwaltungskategorien organisiert und sinnvoll voneinander abgegrenzt.

Die Vor- und Nachteile halten sich bei diesem FreeNAS-Managementsystem in Waage, weshalb es mit der Note *ausreichend* bewertet wird.

5.3.8 EuroNAS Premium

Das EuroNAS Premium ist ein Serverbetriebssystem, welches vorinstalliert auf einem IDE Speichermodul ausgeliefert wird. Die hier betrachtete Premiumversion bietet ein umfangreiches Repertoire an NAS-Diensten, die in mittleren bis großen Unternehmen zum Einsatz kommen. Hauptsächlich konzentrieren sich die Aufgaben des Systems auf die Optimierung von Redundanz, Sicherheit, Kapazität und Geschwindigkeit. Zusätzlich lässt sich dieses System als iSCSI Target konfigurieren, so dass die Speicherelemente des Storagesystems auch im SAN genutzt werden können. Die Verwaltung des Storagesystems erfolgt aus-

schließlich über ein webbasiertes Verwaltungstool, was in den folgenden Punkten näher betrachtet wird.

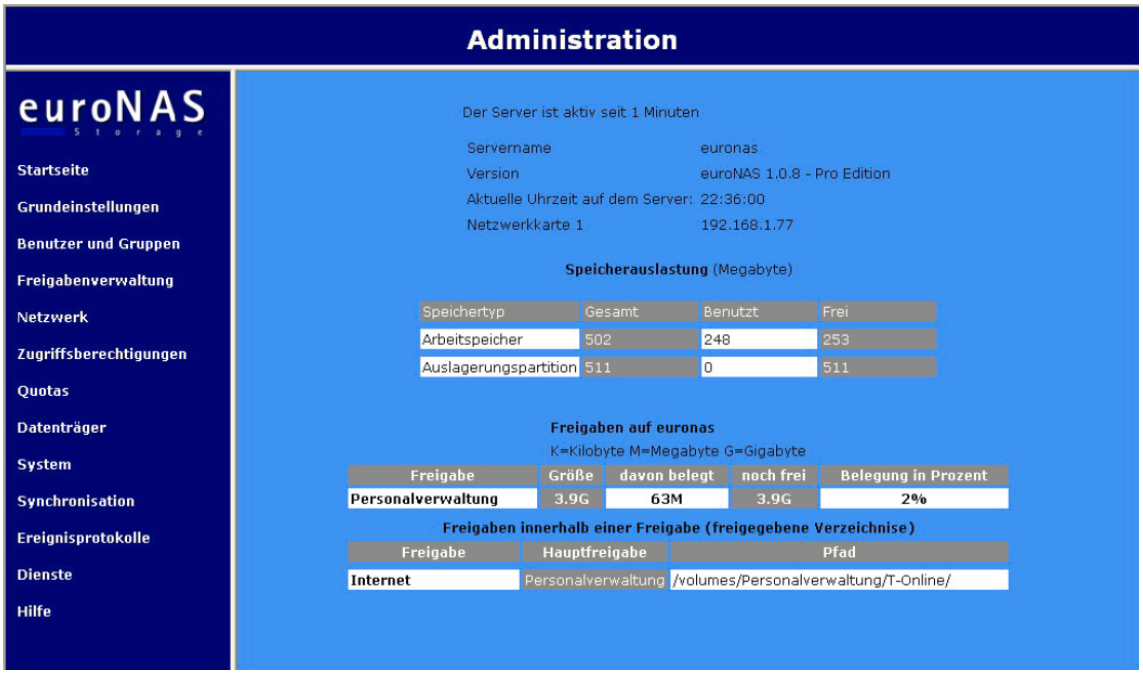

Abbildung 5.8: Die Startseite des EuroNAS Managementsystems (Quelle: www.euronas.de)

- **Technik** Das Verwaltungstool ist bei Auslieferung bereits auf Serverseite vorkonfiguriert, so dass eine nachträgliche Installation oder Konfiguration nicht mehr notwendig ist. Auf Clientseite kann ein beliebiger Browser genutzt werden. Für eine spätere Erweiterung der Hardware in einem System ist dieses ohne Probleme skalierbar.

- **Benutzerfreundlichkeit** Aufgrund der einfachen Seitengestaltung und der vertikalen Navigation auf der linken Seite ist für den Benutzer eine intuitive Bedienung des Systems möglich. Er meldet sich mit einer Administrator-Rolle am System an und kann mit relativ wenig Einarbeitungsaufwand alle notwendigen Änderungen vornehmen. Die Darstellung der Navigationstiefe ist nicht zufrieden stellend. Nach einem Sprung in die dritte Unterseite bekommt der Benutzer dies nicht explizit angezeigt. Auf die Darstellung des Fortschritts sowie der Diagramme wurde bei dieser Anwendung ganz verzichtet. Ein separater Hilfebereich unterstützt den Benutzer bei Fragen, im Kontext eingebettet existieren jedoch keine Hilfen oder Assistenten. Eine ausführliche Hilfe wird mit dem Handbuch im Lieferumfang angeboten.

- **Design** Obwohl das Design auf den ersten Blick altmodisch und langweilig erscheint, ist die Anwendung klar und übersichtlich strukturiert. Das Layout ist einheitlich auf jeder Unterseite, die Menge der Inhalte ist angemessen groß und die Schriftfarben wurden so gewählt, dass jeder Text gut lesbar ist. Diese schlichte und einfache Darstellung erleichtert dem Benutzer das Navigieren durch die Anwendung.

- **Inhalt** Die Verwaltungsobjekte wurden inhaltlich sauber abgegrenzt, indem sie in Kategorien eingeteilt wurden. Die Einstellungsmöglichkeiten der einzelnen Objekte sind dafür ausreichend häufig vertreten.

Die Bewertung der Kriterien befindet sich im Bereich *ausreichend* bis *schlecht*, was in der Gesamtbewertung die Note *schlecht* ausmacht.

5.3.9 Nimbus Halo Manager

Die Managementsoftware Halo Manager liegt in der dritten Version vor und wird mit jedem Storagesystem der Firma Nimbus ausgeliefert. Zu den Storagesystemen gehören das Breeze 10G und das Breeze MX4. Beide sind erweiterbar auf eine Gesamtkapazität von 1.5 bis 55 Terrabyte. Das Managementsystem besitzt eine webbasierte Oberfläche für die Konfiguration des Storagesystems.

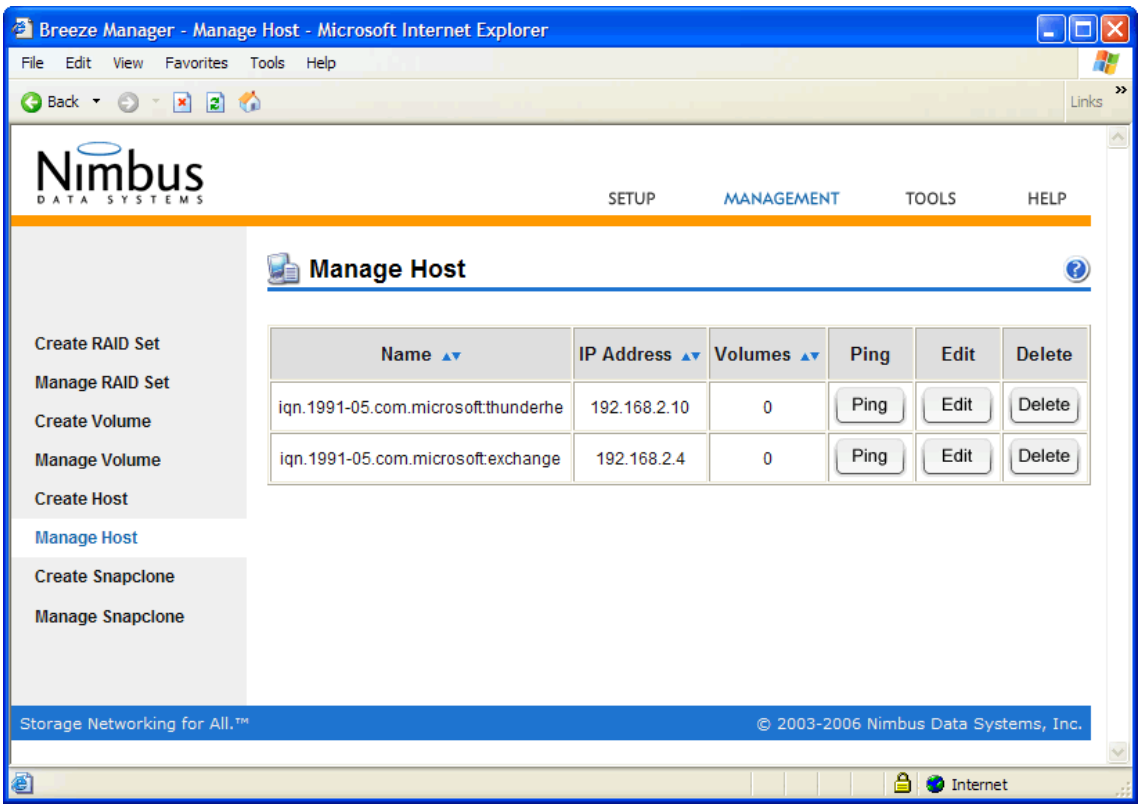

Abbildung 5.9: Die Nimbus Oberfläche bei der Verwaltung eines RAID Sets (Quelle: www.nimbusdata.com/images/products/screenshots)

- **Technik** Bei der Auslieferung ist die Software bereits vollständig installiert, so dass der Administrator keinen weiteren Aufwand mehr hat. Er kann jeden weiteren Konfigurationsschritt über das Webinterface tätigen. Das Webinterface ist serverseitig installiert und kann von jedem JavaScript-fähigen Browser aus bedient werden. Das System ist beschränkt skalierbar. Eine asynchrone Kommunikation wird für die Anzeige des Systemstatus benötigt und eingesetzt.

- **Benutzerfreundlichkeit** Der logische Aufbau der Webanwendung ermöglicht dem Benutzer eine intuitive Bedienung. Die Seiteninhalte sind klar und deutlich von der Navigation abgegrenzt. Der Benutzer bekommt zudem angezeigt in welchem Bereich der Anwendung er sich gerade befindet. Das erleichtert ihm die Einarbeitung und führt ihn direkt zu seinem Ziel. Zudem befindet sich hinter jeder Überschrift einer Kategorie ein Link zu der kontextabhängigen Hilfe. Ein eigenständiger Hilfebereich existiert zusätzlich in der Anwendung. Die Benutzerverwaltung des Managementsystems erfolgt mit einem Rollenkonzept, bei dem mehrere Admininistratoren eingerichtet werden können. Für den aktuellen Systemstatus wurde im Bereich „My Dashboard“ ein Balken- und zwei Zeigerdiagramme eingebaut, die die aktuelle Speicher- und CPU-Auslastung darstellen.
- **Design** Die gesamte Verwaltungsumgebung besitzt ein sehr gutes Design. Die Seiten sind klar und übersichtlich strukturiert, die Farben wurden gut aufeinander abgestimmt und auch Schrifttyp und Schriftfarbe passen sehr gut in das Gesamtbild. Das Aufteilungsschema der Konfigurationselemente bleibt überall gleich. Überfüllte Seiten werden vermieden.
- **Inhalt** Inhaltlich wurde das Managementsystem auf die wesentlichen Elemente beschränkt. Die Möglichkeiten der Konfigurationen könnten noch tiefgründiger sein, und mehr Flexibilität anbieten. Die Inhalte wurden sinnvoll nach ihren Aufgaben zusammengefasst, wie zum Beispiel im Bereich Management die Aufgaben „Create Host“ und „Manage Host“ für die Erzeugung und Verwaltung von iSCSI Hosts.

Es ist eindeutig das Erscheinungsbild gegenüber dem Nutzer, was die Vorteile des Systems darstellen. Die Unterstützung durch Hilfen könnte noch verbessert werden, die Menge des Inhaltes ist zu gering. Die Benotung ist *ausreichend*.

5.3.10 EqualLogic PS Serie

Mit der PS Serie präsentiert die Firma EqualLogic ihre derzeitig aktuellen Speicherlösungen aus dem Enterprise SAN Bereich. Die Geräte können miteinander gekoppelt werden, so dass Hochleistungs-Speicherlösungen von 1,75 bis über hundert Terrabyte erstellbar sind. Die Verwaltung erfolgt nach einem Gruppenschema. Jedes Gerät gehört einer Gruppe an und kann zentral oder individuell verwaltet werden. Das Storagesystem wird komplett mit Betriebssystem und Verwaltungstools ausgeliefert.

- **Technik** Im Lieferumfang befindet sich ein kleines Programm mit dem Namen *Remote Setup Wizzard*. Dieser Assistent übernimmt die Grundkonfiguration eines neuen Systems, wie zum Beispiel das Einstellen der IP Adresse oder das Definieren der Gruppenzugehörigkeit. Nach der Grundkonfiguration steht eine Webanwendung für weitere Konfigurationsschritte zur Verfügung, die über das HTTP-Protokoll kommuniziert. Die gesamte Anwendung basiert auf Java und die graphische Oberfläche wird durch ein Applet im Browser realisiert. So kann eine asynchrone Kommunikation zwischen Server und Client ermöglicht werden. Da die gesamte Software schon vorinstalliert ist, hat der Benutzer keinen weiteren Installationsaufwand. Für die Nutzung der Weboberfläche genügt ein einfacher Browser, der Java Applets ausführen kann. Eine zusätzliche Installation ist nicht mehr notwendig. Durch die Gruppenverwaltung ist das Managementsystem flexibel bezüglich einer Skalierung der Hardware.

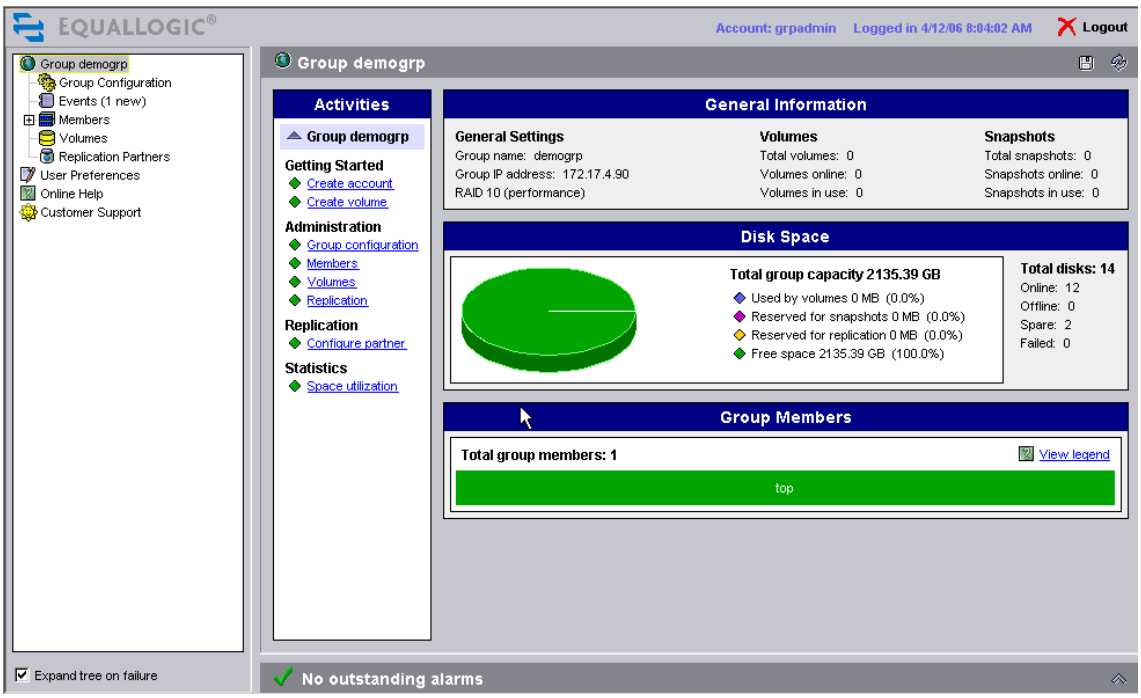

Abbildung 5.10: Die webbasierte Managementoberfläche von Equallogic (Quelle: www.equallogic.com)

- **Benutzerfreundlichkeit** Die klare Trennung von Navigation und Inhaltsfenster gestattet eine intuitive Bedienung durch den Nutzer. Auf den ersten Blick wirkt die große Menge an Optionen zwar erdrückend, nach einer kurzen Einarbeitungszeit ist der Benutzer aber vertraut mit der Navigation. Auch in diesem System muss sich der Benutzer mit einer Rolle anmelden, um seine Berechtigungen zu bekommen. Es können auch mehrere Administratorkonten eingerichtet werden. Das Fehlen der Mehrsprachigkeit und der zusätzlichen Assistenten des Systems sind ein Nachteil dieses Systems. Ein Vorteil ist die ausführliche Hilfe, die in einem separaten Bereich und im Kontext zu finden ist.

- **Design** Das Layout ist mit den vielen Rahmen und Balken verwirrend. Es wurde Wert auf eine einheitliche Struktur gelegt, die in allen Unterseiten wieder zu finden ist. Einige Seiten enthalten sehr viel Inhalte, so dass das Lesen erschwert wird. Die Farb- und Schriftwahl wurden gut getroffen, alle Texte sind gut lesbar.

- **Inhalt** Der inhaltliche Aufbau richtet sich an den Gruppen aus. Deshalb wird auch die Navigation über die Gruppen gesteuert. Die daraus folgenden Einstellmöglichkeiten sind allerdings sehr oberflächlich gehalten worden. Dem erfahrenen Benutzer könnten mehr Optionen zur Verfügung gestellt werden.

Auch bei diesem System gleichen sich Vor- und Nachteile aus. Es wird mit *ausreichend* bewertet.

5.4 Vergleich und Schlussfolgerung

Als klare Favoriten der Bewertung haben sich zwei Systeme hervorgehoben. Das waren das StoreView von Xyratex und die Storage System Software von Intel®. Beide haben die schwierige Aufgabe der Darstellung eines hohen Informationsgehaltes in einer graphischen Benutzerschnittstelle durch eine passende Interaktionsmetapher und gute Gestaltungsmethoden hervorragend gemeistert. StorView bietet zusätzlich viele Animationen an, die im Wesentlichen die Navigation erleichtern. Intel® dagegen bietet eine Vielzahl an Assistenten für die vereinfachte Konfiguration. Bei den anderen Systemen wurden entweder die inhaltliche Tiefe oder die Benutzerfreundlichkeit vernachlässigt. Zu nennen wäre noch das System FreeNAS, dessen gutes Design der Unterstützung von Usability-Experten zu verdanken ist. In Tabelle 5.1 werden die wichtigsten Eigenschaften der betrachteten Systeme noch einmal zusammengefasst.

Kriterium / **System**	**Typ**	**Skalierbarkeit**	**intuitive Bedienbarkeit**	**Hilfen**	**Layout & design**	**Inhalt sinnvoll/ ausreichend**	**Gesamtbewertung**
Cisco SN5428	Webanwendung	○	◐	◐	◐	●	◐
Open-e iSCSI Enterprise	Webanwendung	○	●	●	●	○	◐
Xyratex StorView	Webanwendung	●	●	●	●	●	●
LSI Logic MyStorage Management Software	Client/Server-Anwendung	●	◐	●	◐	●	●
Intel ® Storage System Software	Client/Server-Anwendung	●	◐	●	●	●	●
OpenFiler	Webanwendung	○	◐	○	◐	◐	○
FreeNAS	Webanwendung	○	●	○	●	◐	◐
EuroNAS Premium	Webanwendung	○	◐	○	◐	◐	○
Nimbus Halo Manager	Webanwendung	●	●	◐	●	○	◐
EqualLogic PS Serie	Webanwendung	●	◐	◐	◐	○	◐
	Legende:		● gut	◐ ausreichend		○ schlecht	

Tabelle 5.1: Übersicht über die Eigenschaften der untersuchten Systeme

6 Konzeption des neuen Managementsystems

Bisher hat sich Xiranet bei der Zusammenstellung von Storagesystemen auf die Funktionalität der Betriebssystemsoftware und deren Anwendungen konzentriert. Dadurch wurde ein hohes Maß an Zuverlässigkeit und Ausfallsicherheit erreicht. Das zugehörige Managementsystem für die Konfiguration wurde klein und kompakt gehalten, um alle wichtigen Funktionalitäten für die Systemkonfiguration zu unterstützen.

Da diese Ziele weitestgehend umgesetzt wurden, kann an eine Verbesserung des Managementsystems und dessen Frontend gedacht werden. In diesem Abschnitt wird, basierend auf den Anforderungen und der Analyse des alten Managementsystems, ein neues System konzipiert. Sowohl die inhaltliche und technische, als auch die graphische Darstellung soll dabei genauer betrachtet werden.

6.1 Anforderungen an das Xiranet Managementsystem

Die Kriterien aus Kapitel 3.1 bilden wiederum die Grundlage für die Konzeption des neuen Systems und können um einige Punkte, wie beispielsweise die Bewertungskriterien aus Kapitel 5.2, erweitert werden.
Hier noch einmal die Kriterien im Überblick:

- Geringer Ressourcenbedarf
- Hohe Ausführungsgeschwindigkeit
- Geringe Netzwerkbelastung
- Geringer Installationsaufwand für den Administrator
- Schutz des Quelltextes
- Einhaltung der Lizenzbedingungen
- Vielseitige Einsatzfähigkeit
- Hohe Nutzerfreundlichkeit
- Passendes Design
- Dienstqualität der Managementanwendung

Unter Beachtung der genannten Kriterien wird in den folgenden Abschnitten das neue System konzipiert. Die Analyse in Kapitel 5 hat positive und negative Eigenschaften anderer Systeme aufgezeigt, die in die Konzeption mit einfließen werden.

6.2 Analyse des alten Systems

Das bisher eingesetzte System nutzt einen Apache Webserver mit integriertem PHP4 Modul, da diese Kombination am wenigsten Ressourcen benötigt. Das Managementsystem wird zusammen mit dem Betriebssystem vorinstalliert ausgeliefert, so dass der Kunde keine weiteren Installationsarbeiten tätigen muss. Sobald das Storagesystem eine feste IP Adresse bekommen hat, lässt sich das Web-Frontend ansprechen. Eine richtige asynchrone Kommunikation zwischen Server und Client findet nicht statt, denn das HTTP-Protokoll ist zustandslos und beendet die Verbindung nach jeder erfolgreichen Datenübertragung. Deshalb kann eine asynchrone Anfrage nur durch ein *Polling* von Seiten des Clients realisiert werden. Dabei ruft der Client in einem bestimmten Zeitintervall Daten vom Server ab und gibt diese aus. Diese unidirektionale Kommunikationstechnik wird auch als *Remote Programming* bezeichnet. Der Einfluss der Verwaltungsoberfläche beschränkt sich auf ein Storagesystem und ermöglicht keine Gruppenverwaltung. Das Managementsystem wird in seiner Grundstruktur auf allen Xiranet Storagesystemen eingesetzt, da bloß geringfügige Unterschiede in den Konfigurationsmenus existieren. In den folgenden Betrachtungen dieses Abschnittes wird Bezug auf das XAS500 Storagesystem genommen.

Die Gestaltung des Frontends wurde sehr einfach gehalten, damit eine intuitive Bedienung möglich ist. Der Benutzer meldet sich mit der Administrator- oder Monitor-Rolle an und hat anschließend die Oberfläche in englischer und deutscher Sprache zur Verfügung stehen. Die Seitenstruktur ist in ihrem Aufbau klar und übersichtlich dargestellt. Eine auf der linken Seite vertikal angeordnete Hauptnavigation, bestehend aus Haupt- und Unterkategorien, leitet den Benutzer durch die einzelnen Konfigurationsmenüs. Dieses Navigationsgerüst ist auf jeder Unterseite wieder zu finden, wie auch das Aussehen der einzelnen Seitenelemente. Für die lang anhaltenden Systemprozesse informiert das System den Benutzer mit einer durch Polling gesteuerten Statusanzeige über den Fortschritt. Ein Großteil der Unterseiten besitzt kontextbezogene Hilfeinformationen, die durch kleine Popup-Fenster dargestellt werden. Assistenten oder sonstige Unterstützungen existieren allerdings nicht. Ein separater Hilfebereich gibt Informationen zu Nutzung und Support. Obwohl die Menge an darzustellenden Informationen sehr groß ist, wurden die Inhalte gut lesbar dargestellt. Eine angepasste Farb- und Schriftwahl lässt die Seitenstruktur übersichtlich aussehen und erleichtert die Lesbarkeit. Bei der Anmeldung mit der Monitor-Rolle hat der Benutzer Zugang zu allen Seiten der Anwendung, kann jedoch keine Änderungen vornehmen.

Inhaltlicher Aufbau des Frontends

Die Inhalte wurden nach ihren Konfigurationskategorien strukturiert, was die Hauptbereiche Administration, Systemverwaltung, Festplattenverwaltung, iSCSI-Verwaltung, Statistik und Loginformationen sind. Diese sollen im Folgenden genauer mit Verbesserungsmöglichkeiten betrachtet werden.

Der Bereich Administration konfiguriert alle verwaltungstechnischen Eigenschaften des Storage-Managementsystems. Dazu gehören eine Übersicht für Systeminhalte, Zugangsmöglichkeiten, Logging, Systemverwaltung und die Sicherung. Hier wurden oft Ungleichheiten in der Formatierung gefunden, zu geringe Abstände oder auch zu viele Absätze. Abbildung 6.1 zeigt solche Layoutfehler im Bereich Systemverwaltung. Im Abschnitt „Installierte Firmware“ sollten alle Buttons auf einer Zeile liegen. Die darunter befindlichen Felder für den Dateiupload sollten zudem an das Gesamtlayout angepasst sein. Im Bereich „Einstellungen sichern“ wird der Button „Konfiguration erstellen“ in der Kurzhilfe falsch beschrieben, was den Benutzer verwirrt und gegen die intuitive Nutzung dieser Option spricht.

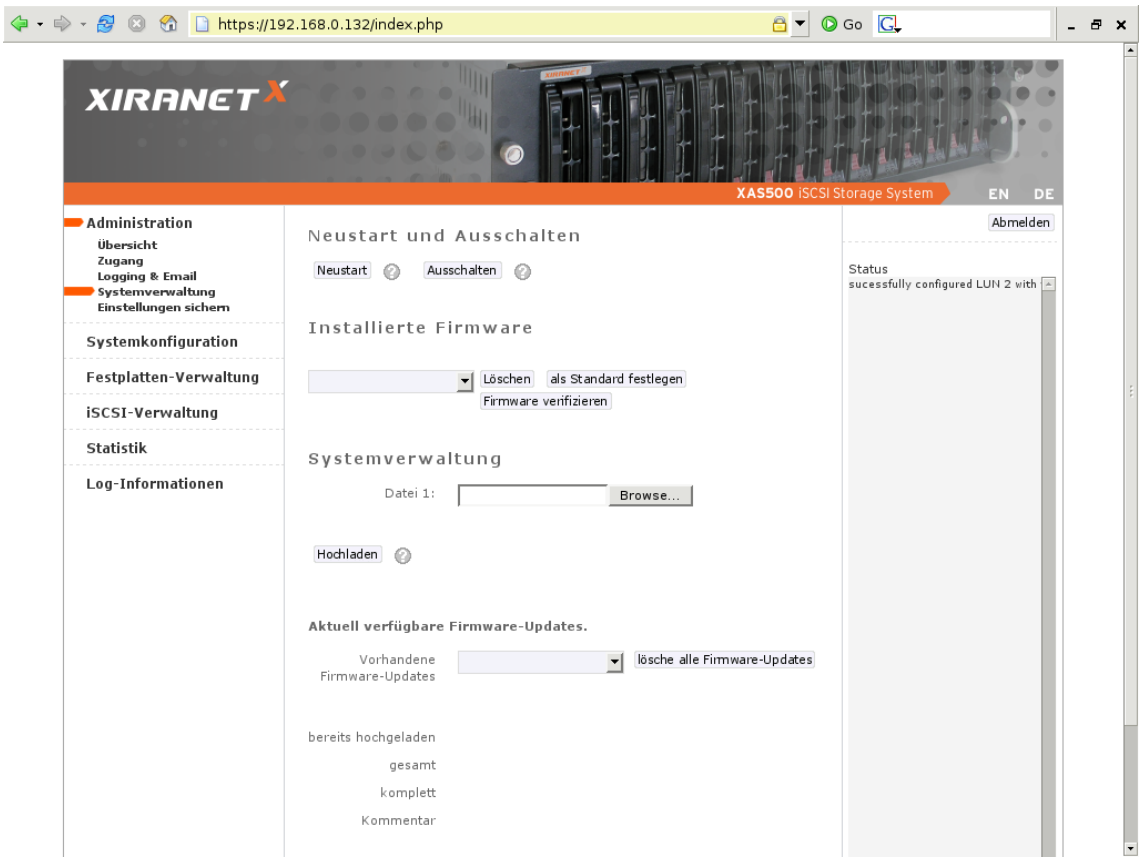

Abbildung 6.1: Die Systemverwaltung des Storage-Managementsystems

Der Bereich „Systemkonfiguration“ bezieht sich nur auf die Grundkonfigurationen des Storagesystems. Dazu gehören die Datumseinstellungen, die Netzwerkschnittstellen und die IPSec-Konfigurationen. Im zuletzt genannten Abschnitt wurde nur ein kleiner Formfehler bei der Beschriftung des NTP-Konfigurationsformulares gefunden.

Der dritte Bereich dagegen ist sehr umfangreich. Er beschäftigt sich mit der Konfiguration der Festplatten des Storagesystemes. Der Unterabschnitt „physische Festplatten“ gibt eine Übersicht über die installierten und konfigurierten Festplatten und deren derzeitigen

Status. Im Unterabschnitt „logische Festplatten“ hat der Benutzer die Möglichkeit, viele physische Festplatten zu einem RAID zusammen zu fügen, indem er den Button „Ändern“ nutzt. Bei den Tests sind hier einige Fehler aufgetreten. Beim Unterbrechen des RAID0 wurde der Verbund als Status „broken“ angezeigt, ein Löschen oder Wiederherstellen wurde jedoch mit einem Fehler quittiert. Beim Öffnen des RAID5 Arrays wurde der Status „degraded“ nicht in der Spalte Status, sondern unter Information ausgegeben. Nach dem Einsetzen der entnommenen Festplatte wurde auch kein Hinweis dafür ausgegeben, dass das System automatisch das RAID Array neu synchronisiert und die Konsistenz des Arrays wieder herstellt. Sobald eine Festplatte aus einem Array entfernt wird, soll diese in der unteren Tabelle „fehlende Festplatten“ angezeigt werden. Diese Funktionalität ist bei beiden Testfällen nicht eingetreten. Zudem ist das +- und –Zeichen in der Spalte Informationen schlecht für die Benutzerfreundlichkeit, was in Abbildung 6.2 zu sehen ist. Dieses Zeichen trägt in Webanwendung häufig die Bedeutung des Hinzufügens von Elementen. Hier jedoch würde der Nutzer nur schwer auf die Idee kommen, dass mit diesem Symbol die Ansicht erweitert wird, bevor er es nicht getestet hat. Erst durch Probieren versteht der Benutzer die eigentliche Bedeutung des Symbols.

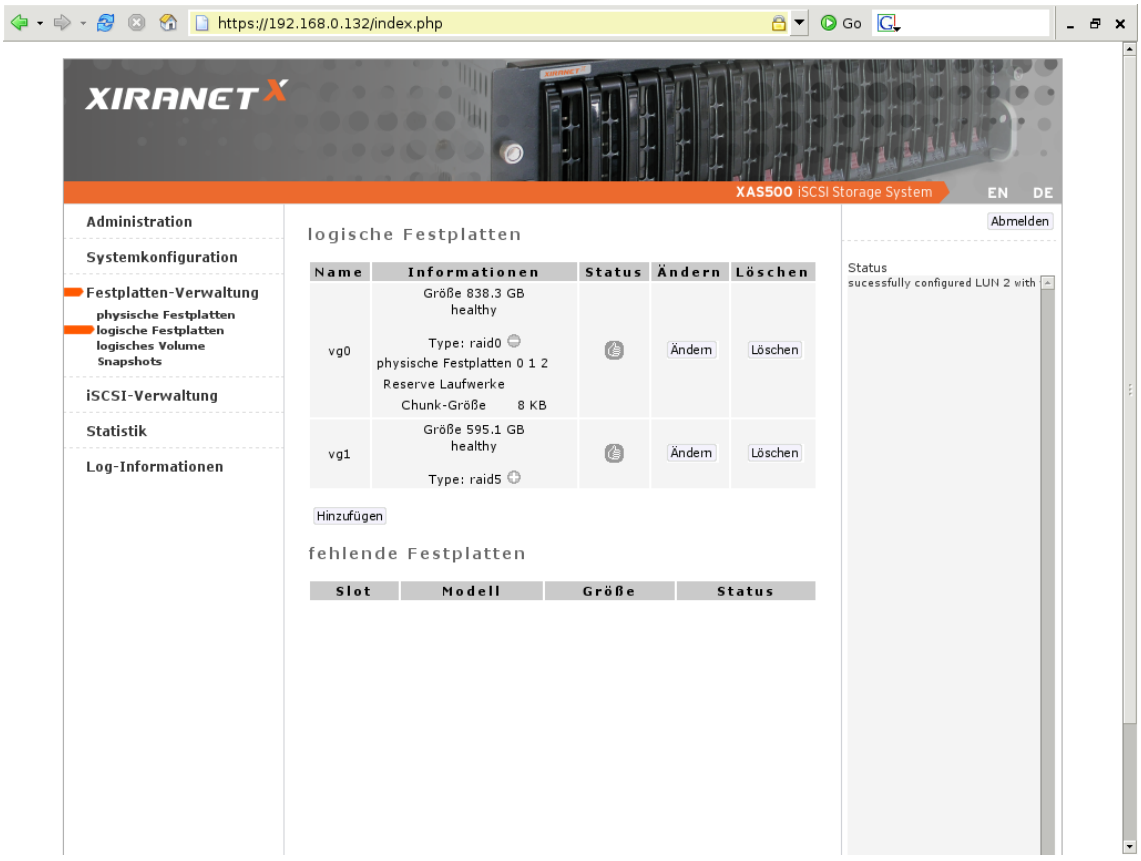

Abbildung 6.2: Menu für das Erzeugen von logischen Festplatten

Der Unterabschnitt „logische Volumen“ dient der Aufteilung einer logischen Festplatte in beliebig viele Teile. Im Menü „Logische Volumen bearbeiten“ trat das Problem auf, dass die Größe des Volumen mit einem Leerzeichen und einer Einheit angegeben werden muss,

ansonsten wird die Zahl mit der Größenordnung Byte interpretiert. Die Fehlermeldung in Abbildung 6.3 wurde ausgegeben, da im Bereich von Byte keine Kommazahlen akzeptiert werden. Auch in der Kurzhilfe wurde nicht darauf hingewiesen, so dass eine intuitive Bedienung an dieser Stelle nicht möglich war. Eine bessere Lösung für das Problem wäre eine getrennte Auswahl der Zahleneinheit, zum Beispiel als Mehrfachauswahlfeld, das den Standardwert Gigabyte festlegt.

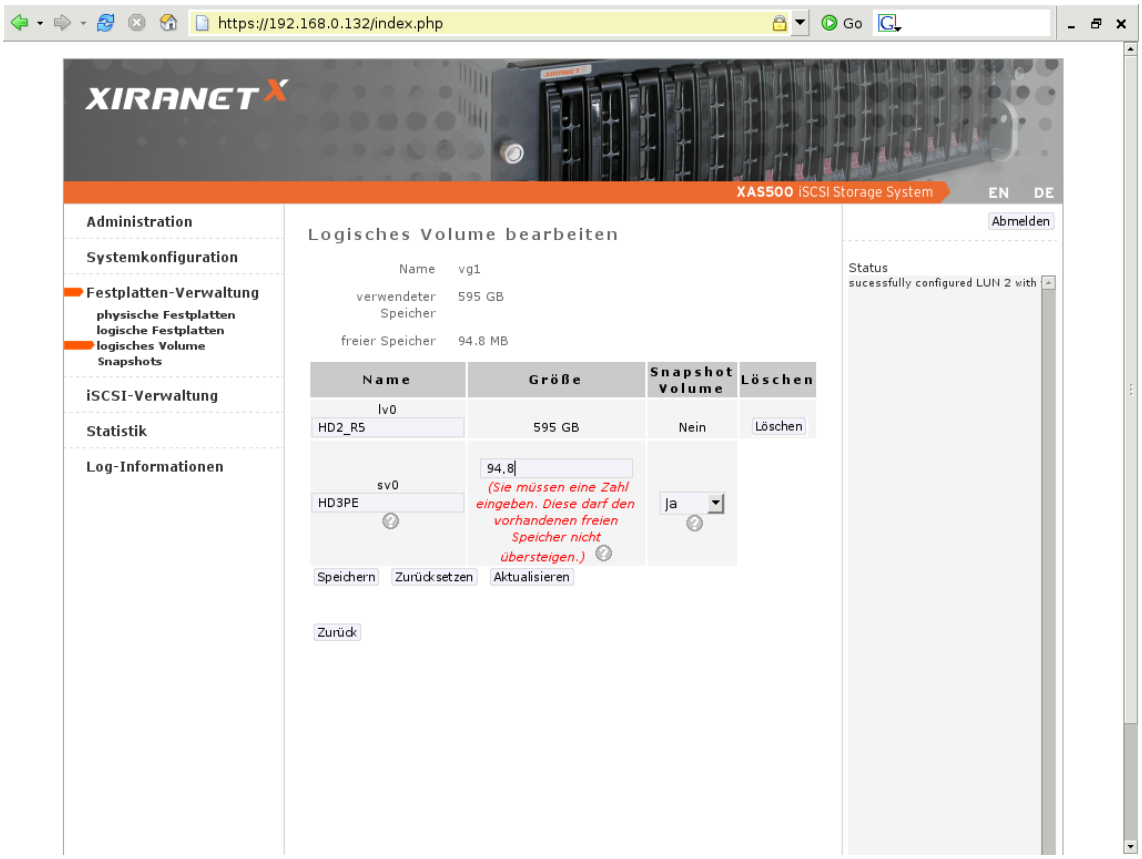

Abbildung 6.3: Die Fehlerbeschreibung beim Anlegen eines logischen Volumen

Im Bereich „iSCSI-Verwaltung“ können die iSCSI-Targets definiert, sowie auch ein iSNS-Server für die domänenweite Verwaltung eingerichtet werden. In diesen Untermenüs stehen dem Benutzer viele Einstellungen zur Verfügung. Bei der graphischen Überarbeitung sollten diese Menüs jedoch übersichtlicher gestaltet werden.

Der Menüpunkt „Statistik“ beinhaltet Systemstatistiken, die der Benutzer mittels Optionenfeldern genauer spezifizieren kann. Darin werden die System- und CPU-Statistiken graphisch in einem Diagramm dargestellt.

Im Menüpunkt „Log-Informationen“ wird eine Logdatei in voller Größe ausgegeben. Diese Darstellung sollte auf jeden Fall überarbeitet werden, um dem Anwender einen besseren Überblick über die Systemereignisse zu gewährleisten.

Weiterhin sind noch einige allgemeine Fehler aufgetreten, die unbedingt verbessert werden müssen. Das Einblenden der Hilfefenster wird derzeit durch serverseitige Dynamik, also einen Neuaufruf der gesamten Seite, realisiert. Dieser Neuaufruf besitzt schon in lokalen

Netzwerken eine relativ hohe Latenz, so dass diese Latenz im WAN-Bereich noch größer werden würde. Das Statusfenster am rechten Rand der Bildfläche, welches permanent sichtbar ist, hat eine zu kleine Sichtfläche. Eine horizontale Darstellung der Statusmeldungen mit eventueller Unterscheidung nach dem Typ der Meldung würde den Informationsgehalt dieser Meldefunktion enorm steigern. Weiterhin existiert keine Sicherheitsabfrage, weder beim Löschen von RAIDs, noch beim Löschen von logischen Volumen.

Technischer Aufbau des Frontends

Beim Aufbau der Webanwendung erfolgte eine Trennung von Anwendungslogik und Design. Eine eigens für die Darstellungszwecke entwickelte XML-Grammatik ermöglicht eine Definition von XML-Templates. Die Erzeugung der HTML-Inhalte aus den Templates wird von der Anwendungslogik durchgeführt und basiert auf der definierten Grammatik, die in der Datei *data.xml* gespeichert wurde und einer Relax NG Schemabeschreibung unterliegt. Alle Templatedateien beginnen mit der Abkürzung *templ_* , gefolgt von dem Namen des jeweiligen Bereiches.

Die darzustellenden Informationen werden vom Management Deamon in einer XML Datei namens *store.xml* geliefert. Sie muss bei jedem Seitenaufruf neu geladen werden, damit eventuelle Änderungen auch sichtbar werden. Das XML-Dokument wird dabei von einem Parser eingelesen und in einen speziell angepassten Objektbaum überführt. Dieses Objektgerüst erlaubt das gezielte Auffinden von Knoten mit Informationen. Da aber häufig nicht der gesamte Objektbaum für die Anzeige einer Seite des Systems benötigt wird, könnte an dieser Stelle durch Einschränkung auf den relevanten Teilbaum die Abarbeitungszeit verkürzt werden. Die Anwendungslogik besteht aus einem komplexen Objektmodell, welches funktional in *packages* eingeteilt ist. Jeder Seitenaufruf wird über die Datei *index.php* ausgeführt, welche das Objektgerüst erzeugt und die Seiteninhalte dynamisch zusammensetzt.

Jede Änderung an dem System, die gespeichert werden soll, wird in die Quelldatei *store.xml* geschrieben und als geändert vermerkt. Das dadurch neu erzeugte XML-Dokument muss ebenfalls der Grammatik aus der Datei *data.xml* entsprechen und deshalb geprüft werden. Anschließend wird der Management-Deamon von der Änderung in Kenntnis gesetzt, damit er die geänderte Datei neu einliest und verarbeitet.

6.3 Inhaltliche Überarbeitung

Im Rahmen des Redesigns des Managementsystems wurde auch die inhaltliche Struktur überarbeitet. Deswegen wurde zu Beginn dieser Arbeit ein Pflichtenheft erstellt, welches die Einstellmöglichkeiten für alle Storagesysteme beschreibt. Aus diesem Pflichtenheft konnte auch ein Netzdiagramm abgeleitet werden, das die Zugehörigkeit der Einstellungspositionen zu den Kategorien ausdrückt. Es kann als vorbereitendes Verzweigungsdiagramm gesehen werden, da darin die Grobstruktur der umzusetzenden Webanwendung ersichtlich wird. Abbildung 6.4 zeigt das Netzdiagramm mit allen Inhalten, die in den nächsten Versionen des Managementsystems realisiert werden.

Für die prototypische Implementierung des neuen Managementsystems wird die inhaltliche Struktur des XAS500-Managementsystems als Grundlage genutzt und geringfügig abgewandelt. Die Konzeption bezieht auf jeden Fall die Möglichkeit der Gruppenverwaltung mit ein. Das bedeutet, dass die Navigation der Oberfläche seinen Einstiegspunkt auf einer höheren Ebene hat. Direkt nach dem Login befindet sich der Benutzer auf der obersten Ebene und betrachtet global das gesamte System. In Abbildung 6.5 wird die Navigationsmöglichkeit des Benutzers graphisch dargestellt.

Durch den Einsatz von XAS1000 Storagesystemen können viele Module zu einem großen Storagesystem verknüpft werden. Eine Zelle, die niedrigste Einheit in der Hierarchie, kann maximal zwei Controller und zwölf JBODs beinhalten. In einem Cluster werden beliebig viele Zellen miteinander verknüpft. An einem Standort können sich mehrere Cluster befinden und aus der Ebene Welt sind alle Standorte sichtbar.
Der in Abbildung 6.5 rot gezeichnete Pfad ist der Hauptweg der Navigation, auf dem der Benutzer in jeder niederen Ebene Informationen über den relevanten Teil des Gesamtsystems angezeigt bekommt. Dieser als hellgrau markierte Bereich dient der globalen Verwaltung des gesamten Systems.

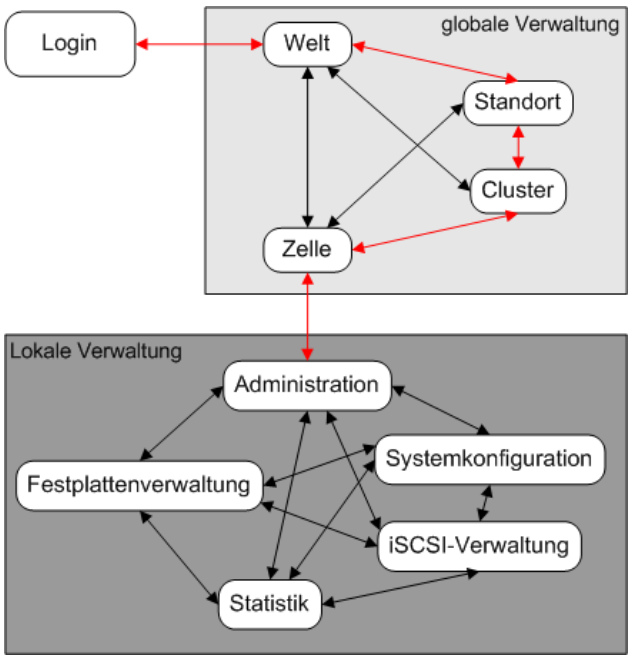

Abbildung 6.5: Die Navigationsstruktur in einem Managementsystem

Im Gegensatz dazu bezieht sich die lokale Verwaltung auf einen modularen Teil des Gesamtsystems, der sich in einer Zelle befindet. Das können ein oder zwei Controller vom Typ XAS1000 sein, die bis zu sechs JBODs ansteuern. Für die prototypische Realisierung dieser Verwaltungsebene werden die Kategorien des XAS500 Storagesystem beibehalten. Die einzige Änderung betrifft den Bereich „Log-Informationen“, der aus den Hauptkategorien herausgenommen und als globale Informationseinheit geführt wird. Das schmale Status-

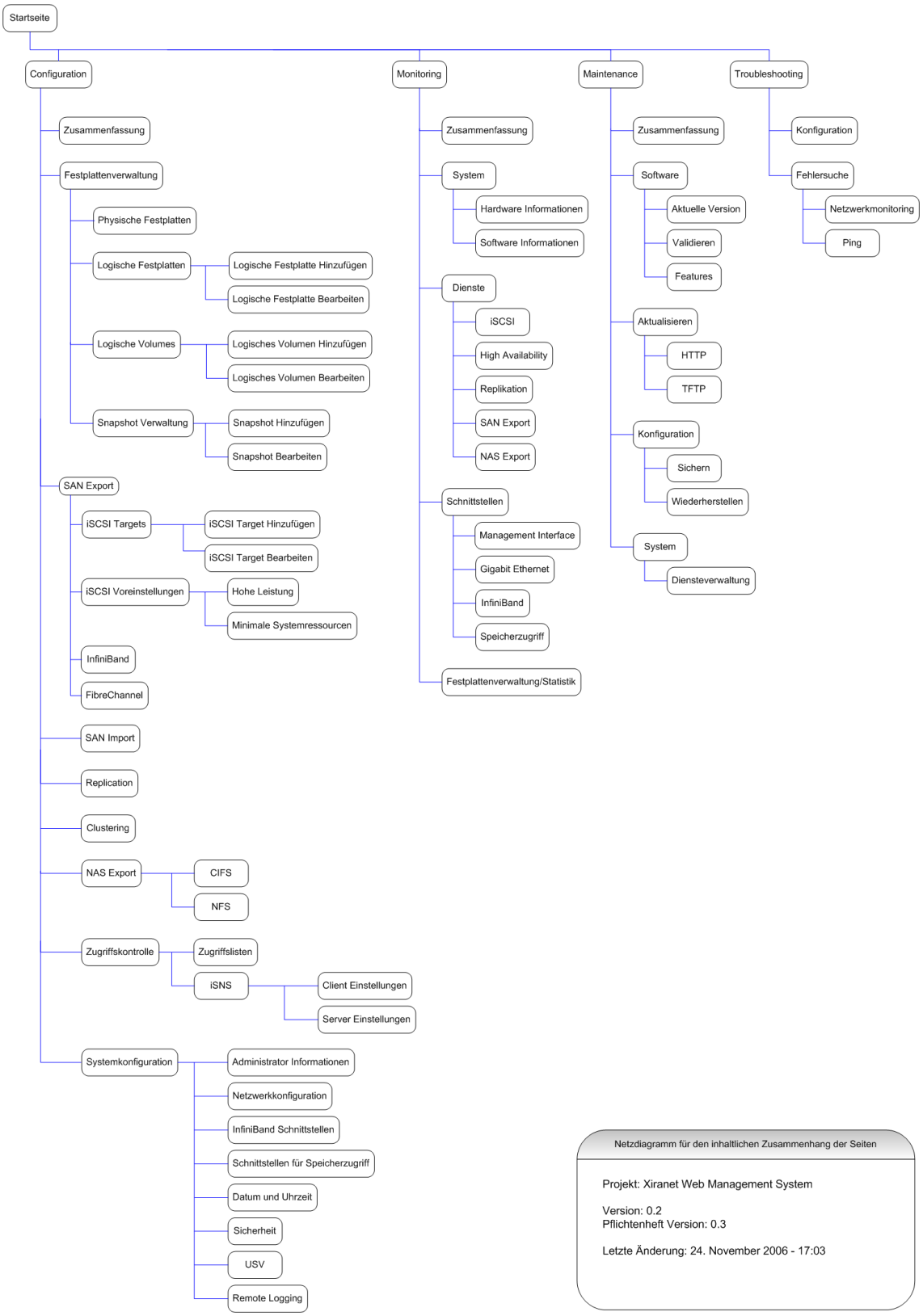

Abbildung 6.4: Strukturdiagramm des zukünftigen Storage-Managementsystems

fenster im rechten Drittel der Seite wird für kontextsensitive Hilfen frei gelassen. Statt dessen werden die Statusinformationen durch einen separaten Balken am unteren Fensterrand permanent angezeigt. In einem ausfahrbaren Fenster werden die gesamten Systemmeldungen zeitlich sortiert angezeigt. Eine Filterfunktion auf dem Statusbalken unterteilt die Anzeige in Fehler-, Warnungs- und Informationsmeldungen ein. Dem Benutzer stehen zwei Ansichten für das Statusfenster zur Verfügung. Die Gesamtansicht erstreckt sich über die gesamte Fensterhöhe, für die Teilansicht kann er selbst die Anzahl der anzuzeigenden Zeilen festlegen. Der Benutzer wird durch visuelle Effekte auf das Eintreffen einer neuen Statusmeldung hingewiesen.

6.4 Technische Überarbeitung

Wie in Kapitel 3.6 eruiert wurde, eignet sich als Basistechnologie für die Realisierung des Managementsystems ein Apache Webserver mit PHP Modul. Für diese Grundlage wird im folgenden Abschnitt die Auswahl des passenden Frameworks beschrieben.

6.4.1 Auswahl des passenden PHP-Frameworks

Gesucht wird ein Framework, welches die Entwicklung der Webanwendung bestmöglich unterstützt. Es werden sechs PHP5-Frameworks untersucht, die alle nach der *Model-View-Controller*-Architektur aufgebaut sind. Die folgende Auflistung zeigt die Hauptentscheidungskriterien und ihre Bedeutungen.

- **Lizenzbedingungen** Es kommen nur Frameworks in Frage, die eine verträgliche Open-Source-Lizenz besitzen. Alles andere ist für den kommerziellen Einsatz unbrauchbar.

- **Community** ist der Oberbegriff für die zu dem Framework gehörende Benutzer- und Entwicklergemeinde. Umso größer dieses Umfeld ist, desto höher ist die Sicherheit und Aktualität des Frameworks.

- **Dokumentationsumfang** steht im direkten Zusammenhang mit der Einarbeitungszeit und der Entwicklungsdauer der gesamten Webanwendung.

- **Eigener Codeanteil** sollte so gering wie möglich sein. Das Framework soll so viel Code wie möglich zur Verfügung stellen und die Wiederverwendung des bereits erzeugten Codes unterstützen.

- **AJAX-Funktionalität** wird von einigen Frameworks unterstützt, viele bieten die Möglichkeit der nachträglichen Implementierung von JavaScript-Bibliotheken an. Dies ist ein Muss-Kriterium für die neue Anwendung, da Seitenbereiche asynchron aktualisiert werden sollen.

- **Sprachunterstützung** wird meist mit einer zentralen Wörterbuchdatei pro Sprache realisiert, deren Inhalte wieder verwendbar sind. Dem Entwickler wird dadurch ein größerer Programmieraufwand abgenommen.

- **Testmöglichkeiten** sind bei jedem größeren Projekt notwendig. Zu den derzeit gängigsten gehören die Unit Tests, die modular die programmierten Funktionen auf ihre Richtigkeit prüfen. Das Erstellen der Testskripte ist noch einmal so aufwendig wie das Programmieren des eigentlichen Codes. Damit kann der Programmierer sicherstellen, dass bei einer nachträglichen Änderung des Quellcodes ein auftretender Fehler entdeckt wird.

- **Größe** ist letztendlich ein Kriterium, das nicht vernachlässigt werden darf. Die Größe des Frameworks sollte aber in Relation zu den genutzten Funktionen des Frameworks stehen. Nicht jedes Framework, das besser ist, muss auch gleichzeitig größer sein.

Nach diesen Kriterien wurden die Funktionalitäten einer engeren Wahl von Frameworks miteinander verglichen. Im Folgenden werden die Frameworks kurz erläutert und anschließend in einer Tabelle gegenüber gestellt.

Symfony ist ein Framework für Webanwendungen, welches von der französischen Webagentur *Sensio* entwickelt wurde. Anfangs nutzte die Agentur ihr Framework nur für eigene Anwendungen. Nach drei Jahren wurde es der Öffentlichkeit unter der MIT-Lizenz zur Verfügung gestellt. Mehr Informationen zu diesem Framework werden in Kapitel 6.4.2 gegeben und sind auf der Projektseite unter [18] zu finden.

Das **CakePHP** Framework ist ebenfalls für die Entwicklung von Webanwendungen entworfen worden. Es ist beliebt durch seine einfache und zeitsparende Implementierung. Auch die Community ist gewachsen, was dem Framework eine schrittweise verbesserte Funktionalität gibt. Dabei verfolgt es die selben Ziele wie Symfony, nämlich das Erzeugen von stabilem geprüften Quellcode. Es wird ebenfalls unter der MIT-Lizenz verbreitet und ist flexibel einsetzbar.

QCodo Web Development Framework ist ein Open-Source Entwicklungsframework für Webanwendungen, das unter der MIT-Lizenz steht. Es beinhaltet zwei Teile, den Codegenerator und die QForms Formularumgebung. Der Codegenerator besitzt die Fähigkeit, ein Datenmodell als Basis für die Webanwendung zu nutzen und dieses in PHP-Code zu überführen. Diese Transformation nimmt dem Entwickler viel Programmierarbeit ab, woraus sich kürzere Entwicklungszyklen für die Anwendung ergeben. Mit QForms werden Funktionen für die Interaktion mit Formularen zur Verfügung gestellt.

PAJAJ ist die Kurzform für *PHP Asynchronous Javascript and JSON*. Es ist ein in PHP geschriebenes AJAX-Framework, mit dem auf einfachste Weise ereignisgesteuerte Webanwendungen erzeugt werden können. Das Projekt existiert seit Anfang 2005 und veröffentlich das Framework unter der LGPL-Lizenz.

Zephyr ist ebenfalls ein AJAX-basiertes MVC-Framework. Es zielt auf die einfache Entwicklung großer Webanwendungen mit geringem Zeitaufwand. Derzeit liegt es in der Version Beta 2.0 vor, was darauf hindeutet, dass es noch in der aktiven Weiterentwicklung ist. Es wird ebenfalls unter der LGPL-Lizenz veröffentlicht.

Zend ist ein Open-Source-Framework und Teil des *PHP Collaboration Project* der Firma Zend. Derzeit liegt es in der Version 0.1.5 vom 11.07.2006 vor und kann unter der BSD-Lizenz bezogen werden. Vor jeder Veröffentlichung wird der Quellcode von Zend geprüft und kann deshalb als sicher eingestuft werden.

Das Ergebnis des Vergleichs zeigt Tabelle 6.1. Die Entscheidung fiel auf das Symfony Framework, das alle aufgelisteten Kriterien unterstützt. Es ist zwar um vieles größer als seine Kontrahenten, aber es dominiert dafür mit seinen Vorteilen. Im Abschnitt 6.4.2 wird das Framework vorgestellt und näher auf die Vorteile eingegangen.

Framework / Kriterium	**Symfony Project**	**CakePHP**	**QCodo**	**PAJAJ**	**Zephyr**	**Zend Framework**
Lizenzbedingungen	✓	✓	✓	✓	✓	✓
Community	✓	✓	✓	×	✓	✓
Dokumentationsumfang	✓	✓	×	×	✓	✓
Eigener Codeanteil	gering	hoch	gering	hoch	hoch	hoch
AJAX-Funktionalität	✓	×	✓	✓	×	×
Sprachunterstützung	✓	×	×	×	×	×
Testmöglichkeiten	✓	✓	×	×	×	×
Größe	6.5MB	1.0MB	1.35MB	2.1MB	1.5MB	1.8MB
	Legende: ✓ gut/vorhanden × schlecht/nicht vorhanden					

Tabelle 6.1: Die Entscheidungskriterien im Überblick

6.4.2 Das Symfony Framework

Die Firma Sensio ist eine Webagentur, die seit dem Jahr 1999 besteht und Webseiten für große französische Firmen entwickelt. Fabian Potencier, Gründer der Firma und Entwickler des Symfony Frameworks, ist an einer weitflächigen Verbreitung des Frameworks interessiert. Durch die Veröffentlichung des Frameworks unter einer Open-Source-Lizenz besteht seiner Meinung nach die Möglichkeit, der Community etwas zurückzugeben und gleichzeitig neue Kunden anzuziehen. „Je mehr Leute mit Symfony entwickeln, desto mehr Firmen werden Symfony adoptieren.“ (Fabien Potencier, [19]).

Das Framework basiert auf dem Mojavi-Projekt, das eine gute Grundfunktionalität bietet. Für die Realisierung des *Models* aus dem MVC-Modell wird das Projekt *Propel* eingesetzt, welches auch *Phing* benötigt. Propel ist ein kleines PHP5 Framework für das Abbilden von Objekten in relationalen Datenbanken, was auch als *object relational mapping, kurz ORM,* bezeichnet wird. Mit der Hilfe von Phing-Bibliotheken wird aus dem Datenmodell des XML-Schema ein Klassenmodell erzeugt. Für die Konfiguration des Symfony Frameworks wird primär das Format *yaml* verwendet. Yaml ist das Akronym für *Yet Another Markup Language* und ist eine Sprache zur Datenserialisierung, ähnlich XML. Mit der einfachen Formatierung von Hashlisten, Listen und Einzelwerten ermöglicht *yaml* die klare und übersichtliche Darstellung von Informationen in Textdokumenten. Weiterhin können auch XML-Dateien zur Konfiguration angewendet werden, die aber mehr Zeit zum Einlesen in Symfony benötigen. Außerdem werden zahlreiche Pear-Erweiterungen verwendet,

weshalb die Pear Standardbibliotheken von PHP5 vorausgesetzt werden. Ab PHP-Version fünf besteht die Möglichkeit, die Paradigmen der objektorientierten Programmierung zu nutzen, wie es auch in Java der Fall ist. Deshalb wurde Symfony ausschließlich für die PHP-Version fünf geschrieben.

Die Komponenten des Frameworks wurden aufeinander abgestimmt und getestet. Jede Erweiterung des Quellcodes wird vor seiner Veröffentlichung ausführlich geprüft und nach seiner Freigabe in der umfangreichen Dokumentation festgehalten. Die zweite Person aus dem Kernteam, Francois Zaninotto, beschäftigt sich ausschließlich mit der Dokumentation des Frameworks. Das derzeit in englischer Sprache vorliegende Benutzerhandbuch wird im Januar 2007 als Buch veröffentlicht. Zudem existieren auf der Homepage des Projektes[1] ein Wiki-System, eine API Dokumentation, das Handbuch in PDF-Form, Quellcodeauszüge, Foren und ein Anwendungsbeispiel mit ausführlicher Beschreibung. Durch die enge Zusammenarbeit mit der Community werden die Entwickler aktiv unterstützt und über die Anwenderbedürfnisse in Kenntnis gesetzt. Solch eine offene Kommunikation wäre durch eine geschlossene Lizenz stark gehemmt. [19]

Das MVC Modell

Das Model-View-Controller-Entwurfsmuster teilt eine Anwendung in die Elemente Datenmodell (*model*), Präsentationslogik (*view*) und Programmsteuerung (*controller*). Diese Trennung soll großen Anwendungen eine übersichtliche Struktur geben und die Wiederverwendung von Quellcode ermöglichen, und damit auch die Flexibilität bezüglich der Erweiterbarkeit erhöhen. Abbildung 6.6 wird das MVC-Paradigma etwas näher erläutern.

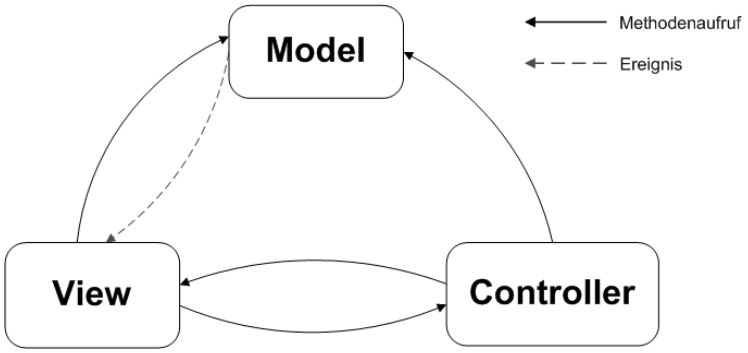

Abbildung 6.6: Schematischer Zusammenhang des MVC-Modells (nach [20])

Das **Modell** hat die Aufgabe der Verwaltung von Anwendungsdaten. Bei einem Methodenaufruf kann der Zustand des Datenmodells abgerufen oder geändert werden. Das Modell ist aber unabhängig vom Controller, weshalb auch keine rückwärtige Beziehung besteht. Der **Controller** ist die Steuereinheit der Anwendung. Er nimmt Benutzereingaben entgegen und verteilt diese auf Funktionen des Modells und des Views. Der Controller legt fest, welches View-Objekt an den Benutzer zurückgegeben wird und teilt dem Modell mit, welches View-Objekt über eine Änderung am Modell informiert werden muss. Der **View**

[1] www.symfony-project.com

ist die graphische Repräsentation des Modells. Zu jedem View kann genau ein Modell zugeordnet werden. Das Modell löst ein Ereignis am zugeordneten View aus, wenn sich sein Zustand geändert hat. Sonst greift der View mit einem Methodenaufruf auf das Modell zu, um die Ausgabedaten anzufordern. Das Modell existiert unabhängig von Controller und View. Es muss keine Kenntnis über den aufrufenden Controller oder View haben. Das macht die Entwicklung und Erweiterung der Anwendung flexibler. Nicht jede der abgebildeten Kommunikationsbeziehungen muss zwingend umgesetzt werden. Dies ist stark von der Implementation abhängig.

Der Aufbau einer Symfony-Webanwendung ist klar strukturiert. An oberster Stelle steht ein **Projekt**, welches mehrere **Anwendungen** beinhalten kann. Diese Anwendungen besitzen **Module**, die die jeweilige Anwendung in funktionale Teileinheiten gliedert. Jedes dieser Module besitzt Aktionen, die eine logische Abarbeitung ausführen und anschließend ein Template für die Ausgabe aufrufen.

Symfony nutzt das MVC-Paradigma für die Entwicklung von Webanwendungen. Es sieht vor, dass eine Trennung von Anwendungslogik, Darstellungslogik und Datenhaltung durchgeführt wird. Wie die Interaktionen bei Symfony verlaufen, zeigt Abbildung 6.7, die eine Abwandlung der Abbildung 6.6 ist.

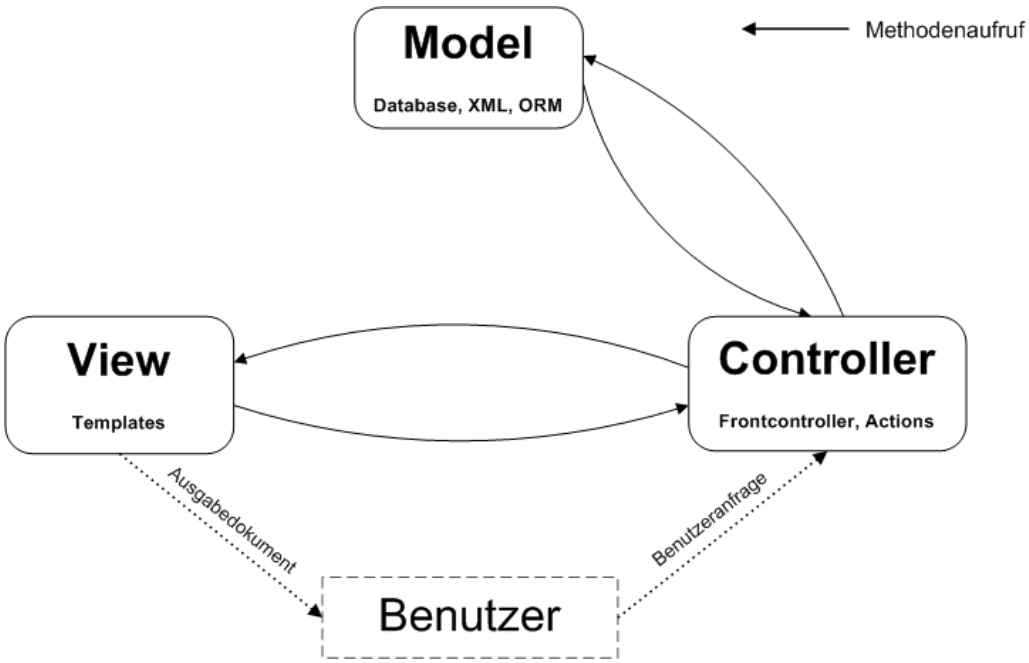

Abbildung 6.7: Das MVC-Modell von Symfony

Das Symfony MVC-Modell spezifiziert einen Frontcontroller pro Anwendung. Der Frontcontroller ist eine *index*-Datei, die alle Benutzerinteraktionen entgegennimmt und weiterverarbeitet. Weiterhin gehören auch die `Actions` zu den Controller-Klassen. Bei der Weiterverarbeitung wird die Anfrage an die adressierte `Action` weitergeleitet. In der jeweiligen `Action` wird die Anwendungslogik ausgeführt, die sich mit der Datenbeschaffung aus dem

Modell beschäftigt und am Ende der Ausführung ein `View`-Objekt aufruft. Der `View` ist für die Ausgabe der Informationen aus dem Modell kombiniert mit XHTML-Quelltext verantwortlich. Bei der Ausgabe wird das fertig generierte XHTML-Dokument an den Benutzer zurückgesendet. Es existiert keine direkte Verbindung zwischen den View-Objekten und dem Modell, wie es im klassischen MVC-Modell vorgesehen ist. Der View erhält seine Daten direkt von der Action, die ihn aufruft.

Damit ist in Symfony das grundlegende Ziel eines MVC-basierten Frameworks, nämlich die Trennung von Inhalt, Layout und Logik, durchgeführt worden.

Die Verzeichnisstruktur des Frameworks

Prinzipiell kann Symfony auf zwei Arten installiert werden. Entweder man nutzt die automatische Installation mit der Unterstützung von Pear, oder man installiert es manuell durch Downloaden und Entpacken der aktuellen Version. Beim manuellen Installieren müssen die Konventionen des Frameworks eingehalten werden, damit die Sicherheit der Webanwendung gewährleistet ist. Näheres dazu ist dem Benutzerhandbuch zu entnehmen. Nach einer erfolgreichen Installation ist eine Verzeichnisstruktur vorzufinden wie sie in Listing 6.1 zu sehen ist.

```
myproject/
    apps/
        myapp/
            ...
    batch/
    bin/
    cache/
    config/
    data/
        sql/
    doc/
    api/
    lib/
        model/
    log/
    test/
    web/
        css/
        images/
        js/
        uploads/
```

Listing 6.1: Verzeichnisbaum eines Symfony-Projekts

Alle Verzeichnisse und Dateien des Frameworks befinden sich im Wurzelverzeichnis `myproject`, wobei `myproject` der Projektname ist. Der strukturelle Aufbau ist dabei klar definiert. Jedes Projekt ist über eine Domain erreichbar, die mit dem Verzeichnis `web` verknüpft ist. So wird sichergestellt, dass die übrigen Verzeichnisse nur über die Indexdatei im Verzeichnis `web` erreicht werden können. Jedes Projekt kann beliebig viele Anwendungen

enthalten, die sich im Verzeichnis `apps` mit dem Anwendungsnamen befinden, im Beispiel von Listing 6.1 steht das Verzeichnis `myapp` für eine Anwendung. Weiterhin kann jede Anwendung aus beliebig vielen Modulen bestehen. Module trennen die Anwendung in funktionale Einheiten auf, die auch eine getrennte Konfiguration besitzen können. Grundsätzlich werden die Konfigurationen auf Projektebene global definiert und können in den unteren Ebenen überschrieben und damit überdefiniert werden. Zudem existieren noch verschiedene Umgebungen, in denen eine Anwendung ausgeführt werden kann. Vordefiniert sind die Umgebungen für Entwicklungs-, Test- und Produktiveinsatz. Jede dieser Umgebung hat andere Konfiguration für Caching, Logging und Fehlerempfindlichkeiten. Es können auch eigene Umgebungen für besondere Einsatzzwecke definiert werden. Das Verzeichnis `batch` ist am Anfang leer und kann für Stapelverarbeitungen genutzt werden, die von der Kommandozeile aus gestartet werden. Das können Prozesse sein, die eine einfache, sich wiederholende Struktur haben.

Im Verzeichnis `bin` befinden sich die ausführbaren Dateien des Frameworks, die für die Administration auf Kommandozeilenebene benötigt werden. Dieses Verzeichnis wird zur Ausführung der Webanwendung nicht benötigt und ist deshalb nicht in einem Produktivsystem zu finden. Das Verzeichnis `cache` wird von jeder Anwendung als temporäre Ablage für vorkompilierte Dateien genutzt. Dank des integrierten Caching-Mechanismus wird die Ausführung der Anwendungen beschleunigt.

Im Verzeichnis `data` werden fixe Daten sowohl vom Framework, als auch von den Anwendungen abgelegt. Das können beispielsweise ein Datenbankschema oder XML-Grammatiken sein. Unter `doc` sollen die Dokumentationen von Framework und eigenen Anwendungen abgelegt werden. Das `lib`-Verzeichnis wurde für fremde Klassen und Bibliotheken eingerichtet. Das Unterverzeichnis `model` ist der Platz für das Objektmodell des Projektes. Im Verzeichnis `log` landen alle Logdateien, die Symfony automatisch erzeugt. Das ist gewöhnlich eine Datei pro Anwendung und Umgebung. Dort können auch Logdateien von Datenbank und Webserver abgelegt werden. Das `test`-Verzeichnis beinhaltet alle PHP-basierten Unittests, die von Symfony beim Anlegen eines neuen Moduls mit angelegt wird. Dabei werden aber nur einfache Testdateien mit ihrem Grundgerüst angelegt, die dann vom Entwickler vervollständigt werden müssen.

```
myapp/
    config/
    i18n/
    lib/
    modules/
        mymodule/
    templates/
        layout.php
        error.php
        error.txt
```

Listing 6.2: Verzeichnisbaum einer Symfony-Webanwendung

Listing 6.2 zeigt die Verzeichnisstruktur, die beim Anlegen einer neuen Anwendung mit Hilfe des Kommandozeilenbefehls `symfony init-app myapp` erzeugt wurde. Das Verzeichnis `config` enthält alle Konfigurationsdateien auf Anwendungsebene. `i18n` beinhaltet die

Wörterbuchdateien für die Internationalisierung, wobei pro Sprache eine XML-Datei erzeugt werden muss. In `lib` können anwendungsspezifische Bibliotheken abgelegt werden. Alle Module werden automatisch im `modules`-Verzeichnis angelegt und besitzen eine festgelegte Grundstruktur. Das `templates`-Verzeichnis enthält die Standardlayouts für die gesamte Anwendung. Die Datei `layout.php` wird bei erfolgreichem Ausführen der Anwendung ausgegeben, `error.php` im Fehlerfall. Die Datei `error.txt` wird mit Fehlernachrichten gefüllt, wenn keine Browserausgabe definiert wird. Dies kann bei Unittest der Fall sein, die von der Kommandozeile aus gestartet werden.

```
mymodule/
    actions/
        actions.class.php
    config/
   lib/
   templates/
      indexSuccess.php
   validate/
```

Listing 6.3: Verzeichnisbaum eines Symfony-Moduls

Die in Listing 6.3 dargestellte Verzeichnisstruktur zeigt den Inhalt eines Moduls, das mit dem Kommandozeilenbefehl `symfony init-module myapp mymodule` automatisch angelegt wird. Als Parameter werden der Modulname und der Anwendungsname benötigt. Im Unterverzeichnis `actions` befinden sich die Controller-Klassen, die als `Actions` bezeichnet werden. In `config` werden die lokalen Konfigurationsdateien abgelegt. Das können die selben Dateien sein, wie auf globaler Ebene. Im Verzeichnis `lib` können alle benutzerdefinierten Bibliotheken und Hilfsklassen abgelegt werden, die nur in diesem Modul angewendet werden. Weiterhin beinhaltet `templates` alle View-Klassen mit einer festen Namenskonvention. Der erste Teil des Namens ist die aufrufende Methode der Actionklasse, der zweite Teil ist das Ausführungsergebnis der Action. `Success` steht für erfolgreich ausgeführt, `Error` für den Fehlerfall. Das Verzeichnis `validate` beinhaltet alle yaml-Dateien, die eine Formularbewertung konfigurieren. In diesen Dateien kann klar definiert werden, welche Werte die Übergabeparameter an eine Action besitzen dürfen und was im Fehlerfall ausgeführt werden soll. Als Namenskonvention ist der Dateiname der Name von der zu bewertenden Aktion.

Weitere Eigenschaften von Symfony

Ein weiteres Ziel von Symfony ist die Wiederverwendung von häufig genutzten Quelltextabschnitten. Es bietet auch eine Vielzahl an Hilfsklassen, die `Helpers`, an, die Vereinfachungen treffen sollen. Im Folgenden werden die wichtigsten Eigenschaften erläutert:

- **simple templating**. Als Templatesystem wird PHP verwendet, das durch die Mächtigkeit der Hilfsklassen unterstützt wird und eine schnellere und effektivere Entwicklung ermöglicht. Es besteht zusätzlich die Möglichkeit, ein sekundäres Templatesystem zu verwenden. In dieser Hinsicht sind dem Entwickler keine Grenzen gesetzt.
- **cache management** ist das Zwischenspeichern von HTML-Quelltextteilen oder ganzen Dateien in einem Cacheverzeichnis, um die Ausführungsgeschwindigkeit zu er-

höhen. Der zu Grunde liegende Mechanismus ist sehr einfach gehalten. Bei einer Benutzeranfrage sucht der Frontcontroller im Cacheverzeichnis nach einem Eintrag für die aufgerufene `Action`. Wenn ein Eintrag vorhanden ist, wird dieser an den View weitergeleitet, und nicht die `Action` ausgeführt. Für dynamische Seiteninhalte ist das Cachen standardmäßig deaktiviert, kann jedoch in den Konfigurationsdateien individuell an die Bedürfnisse der Anwendung angepasst werden. Es besteht auch die Möglichkeit, Teile eines Templates oder Komponenten in das Caching einzubinden.

- **smart URLs** bezeichnet ein völlig neues Routing-System, das in Symfony zum Einsatz kommt. Das folgende Beispiel eines Aufrufes soll das Routingverfahren verdeutlichen:

  ```
  http://myapp.example.com/index.php/article/read/id/100
  ```

  Die Sublevel-Domain verlinkt direkt in das Verzeichnis `/web` des Frameworks. Die Datei `index.php` ist der Frontcontroller, der das Modul `article` mit der Action `read` aufruft. Als Parameter wird der Name `id` mit dem Wert `100` übergeben. Weitere Parameter werden mit dem gleichen Name-Wert-Muster angehängt. Das ist das Standardverhalten beim Auflösen von Adressen in Symfony. Das Verhalten kann auch bei Bedarf modifiziert werden. Dafür muss in der Datei `routing.yml` die Struktur der Auflösung umgeschrieben werden. In den meisten Fällen ist solch eine Änderung aber nicht notwendig.

- **scaffolding** unterstützt den Entwickler bei den Grundaufgaben des Datenbankzugriffes. In Zusammenarbeit mit Propel, einer Bibliothek für den Datenbankzugriff, wird ein Grundgerüst an Actions und Templates erstellt. Dieses übernimmt die Aufgaben Erzeugen, Suchen, Aktualisieren und Löschen von Datenbankeinträgen, auch kurz mit dem Akronym C.R.U.D. bezeichnet.

- **multilingualism and I18N support**. Symfony beinhaltet Mechanismen für die Entwicklung von mehrsprachigen und lokal angepassten Webanwendungen. Die Mehrsprachigkeit einer Webanwendung bezieht sich auf drei Hauptkategorien. Das sind Standards und Formate, die Textinformationen in der Datenbank und die Übersetzung der Sprache in der Oberfläche. Durch die Definition einer Kulturzugehörigkeit wird in dem Sessionobjekt eines Benutzers die Länderkennung für Übersetzung und Land festgelegt. Es muss also zu Beginn der Entwicklung eine Standardkultur definiert werden, die zu einem späteren Zeitpunkt auch noch umgestellt werden kann. Die damit eingestellte Sprache dient als Basissprache für den Übersetzungsmechanismus. Um dies nutzen zu können muss für jede Sprache eine XML-Wörterbuchdatei geführt werden, die ein XLIFF-Format beinhaltet. Der Name dieser Datei trägt dabei immer das Namenskürzel der Zielsprache, also `messages.fr.xml` für die Übersetzung von der Basissprache in die französische Sprache. Die Standards und Formate werden von den Helpers genutzt, die länderspezifische Komponenten zur Verfügung stellen. So erzeugt beispielsweise der Helper `input_date_tag($name, $value, $options)` ein Eingabefeld für das Datum, welches automatisch formatiert wird und einen graphischen Kalender anbietet.

- **AJAX support**. Er basiert auf der Verwendung von JavaScript-Helperklassen, die in PHP geschrieben werden und den JavaScript Quelltext automatisch erzeugen. Diese Klassen bieten eine Vielzahl von Funktionen an, die eine asynchrone Kommunikation erzeugen können. Deshalb wird im Handbuch die Bezeichnung **AJAX-Helpers** verwendet. Die Remotefunktionen können sofort, bei Benutzung eines Formularelements oder in periodischen Zeitabständen aufgerufen werden. Der Unterschied zum normalen Seitenaufruf besteht darin, dass nur ein Zielelement der Seite angegeben werden muss, welches aktualisiert werden soll. Die Rückgabewerte der Ausführung der aufgerufenen Action werden dann direkt in dieses Zielelement eingesetzt.

- **unit testing**. Der Einsatz von Unittests ist das wichtigste Werkzeug für die Entwicklung von großen Softwarelösungen. Der Name kommt von der Vorgehensweise der Tests, da so genannte Programmeinheiten separat getestet werden können. Für die Tests werden Testfälle für die Eingabeparameter definiert und auf die jeweiligen Funktion angewendet. Der Aufwand für das Erstellen solcher Testklassen ist sehr hoch, hat aber den Vorteil, dass die Tests sofort Fehler an ihrem Ursprung aufdecken. Bei einer nachträglichen Änderung von Methoden beispielsweise können sich schnell Fehler einschleichen, die nicht sofort vom Entwickler gefunden werden können. Deshalb wird empfohlen, jede neu programmierte Methode gleichzeitig mit einem Test zu erstellen. An dieser Stelle unterstützt Symfony den Entwickler, indem es beim Erzeugen von Modulen Testklassen für das Testframework *Simpletest* anlegt. Der Entwickler muss diese Klassen nur erweitern und anpassen. Für die Ausführung aller Testklassen stellt Symfony eine separate Testumgebung bereit, die alle Testfälle für eine Anwendung als Stapelverarbeitung ausführen kann. Es können drei verschiedene Arten von Tests durchgeführt werden:

  - **default unit tests** ist die einfachste Form der Tests, bei der jede Methode einer Klasse separat getestet werden kann. Die Testfunktion vergleicht dabei, ob für einen bestimmten Eingabeparameter die richtigen Ausgabeparameter erzeugt werden. Wenn dies nicht der Fall ist, wird eine Fehlerausschrift erzeugt.
  - **Web Browser Session**-Simulation stellt eine Browsernutzung nach. Die einfache Variante davon stellt eine Anfrage an das Framework, ohne dass ein Webserver aktiviert sein muss. Diese Art Test ist sehr schnell. Langsamer dagegen sind die ausführlichen Web-Tests mit dem *WebTestCase*-Objekt. Dieses Objekt simuliert die Interaktion mit Formularelementen und Hyperlinks während des Tests und benötigt einen aktiven Webserver, der die Anfragen verarbeitet.
  - **Selenium** Webtests basieren auf JavaScript und bieten die Möglichkeit, komplexe Interaktionen wie beispielsweise AJAX-Aufrufe zu simulieren. Die Testdateien werden in Form von HTML-Dateien im Verzeichnis `selenium` abgelegt. Jede HTML-Datei definiert die Testaktionen in einer dreispaltigen Tabelle mit Kommando, Ziel und Wert als Parameter in den Zellen der Tabelle. Die gesamte Testsuite kann dann von Clientseite durch den Aufruf der Datei `index.html` im Verzeichnis `selenium` gestartet werden. Diese Version der Tests hat den Vorteil, dass die Interaktionen durch den *Selenium Recorder* Erweiterung für Mozilla Firefox aufgezeichnet werden können und damit die Tests viel schneller erstellt werden können.

6.4.3 Umsetzung mit Symfony

Es wurde beschlossen, das Projekt komplett mit dem Symfony Framework umzusetzen. Da das Storagesystem derzeit die alte Version des Managementsystems nutzt, und dieses den Bedürfnissen nach weiterentwickelt wurde, muss erst einmal der gleiche Entwicklungsstand erreicht werden. Das Managementsystem kann erst dann umgestellt werden, wenn ein gleicher inhaltlicher Stand erreicht ist und das neue Managementsystem fehlerfrei betrieben werden kann.

Im ersten Schritt der Umstellung soll die graphische Oberfläche vom alten Managementsystem weiter verwenden und nur die Backend-Technik umstellen. An dieser Stelle ist es wichtig herauszufinden, ob alle geplanten Umstellungen mit dem Symfony-Framework realisiert werden können. Davon hängt der Erfolg des ganzen Projektes ab. In der Einarbeitungsphase wird das Grundgerüst der Webanwendung aufgestellt. Da Symfony erstmalig eingesetzt wird, muss ein erhöhter Zeitaufwand eingerechnet werden. Das Grundgerüst des Layouts besteht aus einer globalen Layout-Datei, in die die Komponenten und Templates in Abhängigkeit vom jeweiligen View dynamisch eingesetzt werden. Abbildung 6.8 zeigt das Grundlayout der Seite mit seinen Teilkomponenten.

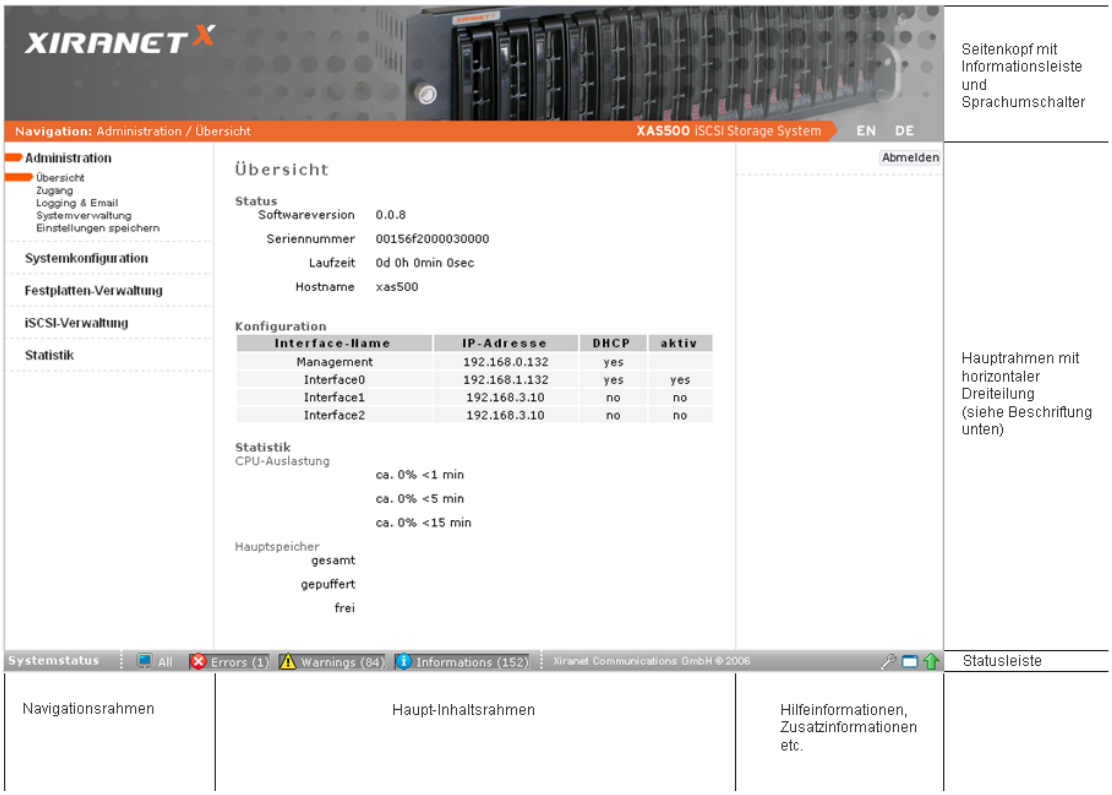

Abbildung 6.8: Strukturelle Aufteilung des Grundlayouts

Der rote pfeilähnliche Balken im Seitenkopf beinhaltet in seinem linken Teil die Pfadanzeige, die Informationen über die aktuelle Position des Benutzers gibt. Diese so genannte „breadcrumb“ und der sich rechts neben dem Pfeil befindende Sprachumschalter wurden als *Component*-Element umgesetzt, was eine Komponente mit Action und Template ist.

Der sich unterhalb des Seitenkopfs befindende Hauptrahmen ist in drei Teile untergliedert. Links das Navigationsfeld, in der Mitte der Inhalt und rechts der Hilfebereich für die kontextsensitive Hilfe und Zusatzinformationen. Jede Aktion eines Moduls gibt ein Template an den View, das in diesen Hauptrahmen des Layouts eingesetzt wird. In diesem Template wird auch die Navigationskomponente auf der linken Seite initiiert. Die Navigation wurde als Partial[2] realisiert, das in das jeweilige Modultemplate eingebunden wird.

Die Modulstruktur wurde nach den funktionalen Einheiten der Oberfläche eingeteilt und ist teilweise auch aus den Hauptkategorien der Navigationselemente zu erkennen. In Abbildung 6.9 werden die Kommunikationsbeziehungen zwischen den Modulen aufgezeigt, die in der folgenden Liste näher erklärt werden:

- **loginModule** Das Modul ist für den Anmeldevorgang verantwortlich. Es erzeugt eine Eingabemaske und wertet die übertragenen Informationen aus. In diesem Kontext werden auch alle wichtigen Initialisierungen für korrekt angemeldete Benutzer durchgeführt wie beispielsweise das Schreiben von Benutzerdaten in den Session-Kontext.
- **navigationModule** Dieses Modul ist der Einstiegspunkt für den Benutzer nach einer erfolgreichen Anmeldung. Die oberste Ebene des Navigationsbaumes ist die globale Betrachtung auf das Gesamtsystem, in der der Benutzer alle Standorte angezeigt bekommt. Nach der Auswahl eines Standortes bekommt er alle Cluster angezeigt, von denen er ebenfalls wieder ein Element auswählen kann, um eine Ebene tiefer zu gehen. In dem Cluster sieht er alle enthaltenen Zellen als Tabelle gelistet. In der Zelle befinden sich Controller und JBODs. Das besondere an diesem Modul ist, dass nur für die Weltansicht ein statischer Seitenaufruf durchgeführt wird. Alle weiteren Aufrufe für die Navigation in tiefere Ebenen wird durch AJAX-Aufrufe realisiert, die nur den Inhalt des Hauptrahmens aktualisieren. Wie die Navigationselemente graphisch gestaltet wurden, wird in Kapitel 6.5 näher erläutert.
- **adminModule** Im Bereich Administration können alle Einstellungen für die Verwaltungsarbeit am System eingestellt werden. So zum Beispiel die Einstellung der Art des Zuganges, des Administratorpasswortes oder der Logging-Strategie.
- **sysconfigModule** Der Bereich Systemkonfiguration bezieht sich auf die spezifischen Einstellungen des Storagesystems. Dies umfasst die Zeiteinstellungen, Schnittstelleninformationen und IPSec-Konfigurationen.
- **diskModule** Die Festplattenverwaltung ist der größte und wichtigste Bereich des gesamten Managementsystems. Bei dem Unterpunkt physische Festplatten werden alle installierten Festplatten aufgelistet. Diese können unter "logische Festplatten" mittels RAID verknüpft werden. Danach besteht die Möglichkeit der Aufteilung dieser logischen Einheiten in virtuelle Festplatten. Zusätzlich können von virtuellen Festplatten Schnappschuss-Kopien angefertigt werden, die ebenfalls verwaltet werden müssen. In allen diesen Untermenüs existiert eine Maske für die Auflistung, das Verändern und Hinzufügen von Geräten. Einige Eingabemasken konnten wegen ihrer Ähnlichkeit übernommen werden.

[2] *Partial* ist eine komponenten-ähnliches Element, das nur Darstellungslogik und keine Anwendungslogik besitzt. Die Ausführungsgeschwindigkeit eines *Partials* ist um vieles höher als die einer *Komponente* oder eines *Moduls*.

- **iscsiModule** In diesem Bereich können iSCSI Targets und -Server verwaltet werden. Auch hier existieren für jedes Untermenü Formulare für das Hinzufügen und Bearbeiten der einzelnen Objekte.

- **statModule** Der Statistik-Bereich dient der Ausgabe von Diagrammen zu CPU, Netzwerk und Festplatten.

- **ajaxData** Dieses Modul hat eine besondere Stellung in dieser Webanwendung. Es besitzt nur Actions, die global von jedem Teil der Anwendung aufgerufen werden können. Da diese auszuführenden Prozesse von AJAX-Aufrufen initiiert werden, dürfen sie kein vollständiges Layout an den Browser zurücksenden. Deshalb werden nur die Inhalte des zugehörigen Templates zurückgegeben, ohne Layout-Gerüst.

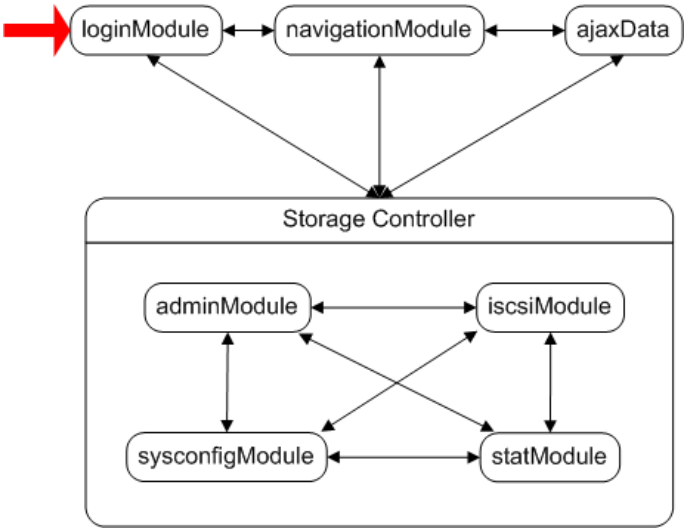

Abbildung 6.9: Die Interaktionsbeziehungen zwischen den Modulen

Nach der Fertigstellung dieses Bearbeitungsschrittes wird der Benutzer nicht bemerken, dass sich die Technologie im Hintergrund geändert hat. Vom Entwickler wird hingegen ein geringer Vorteil in der Ausführungsgeschwindigkeit erwartet. Dieser Vorteil wird aber so gering sein, dass er nur mit einer Laufzeitmessung erkannt werden kann.

Die Neuerungen am Managementsystem

Wie schon in Kapitel 6.3 angesprochen, wurde der Bereich „Log-Informationen“ herausgenommen. In diesem Bereich wurde vorher der Inhalt der Logdatei *xiranet.log* ungefiltert ausgegeben. Die darin befindlichen Ausgaben übertrafen größtenteils die mögliche Anzeigebreite und zwangen den Benutzer zum Scrollen. Zusätzlich befand sich im rechten Drittel der Seite eine Bereich „Status“, der gefilterte Ausgaben dieser Logdatei angezeigt hat. Dieser Statusbereich erfüllte jedoch nicht die nötigen Anforderungen, da die Einträge zu lang waren und neue Statusmeldungen unter die vorherigen gesetzt wurden.

Im neuen Managementsystem wurden beide Statusanzeigen durch eine Statusleiste am unteren Fensterrand ersetzt. Die 21 Pixel hohe Leiste hat dieselbe Breite wie die gesamte Anwendung und ist permanent sichtbar. In der Mitte befinden sich Schalter, die die anzuzeigenden Meldungen nach den Typen Fehler, Warnungen und Informationen filtern. Das Inhaltsfenster der Statusmeldungen ist versteckt und kann mit dem rechts auf der Leiste befindlichen Buttons aus- und eingefahren werden. Der grüne Pfeil erwirkt ein anteiliges Ausfahren, das weiße Fenster hingegen das Ausfahren auf die gesamte Fensterhöhe. Jede neu eintreffende Meldung wird dabei oben in der Liste angezeigt. Jede Meldung wird mit ihrem Typ als Symbol, dem Datum und der Beschreibung angezeigt. Längere Beschreibungstexte werden in dem Tabellenfeld umgebrochen, so dass ein horizontales Scrollen nicht notwendig wird. Zusätzlich existiert noch ein Einstellungsfenster, das mit einem Klick auf das Schraubenschlüssel-Symbol ausgefahren wird. Die darin befindlichen Optionen bewirken das Einschalten des Tabellenkopfes, der visuellen Anzeige über neue Statusmeldungen, der Anzahl der Zeilen, die in der Teilansicht sichtbar sein sollen, und der Anzahl der gesamt anzuzeigenden Statusmeldungen in Zeilen. Das besondere an dem Einstellungsfenster ist der Einsatz der AJAX-Technologie für die asynchrone Abspeicherung der geänderten Werte. Alle vier Formularelemente werden mit einer JavaScript-Funktion überwacht, die bei Änderung eine Action aufruft und die Änderungen sofort durchführt. Beim Einschalten des Tabellenkopfes in der Meldungsliste zum Beispiel wird der gesamte Inhalt mit Kopf in das Fenster geschrieben.

Mit jedem Seitenaufruf wird auch die Statusleiste neu geladen und mit aktuellen Informationen gefüllt. Bei der Benutzung eines Filterschalters sowie zeitgesteuert nach 20 Sekunden wird mit Hilfe von JavaScript eine *Action* abgefragt, deren Rückgabewert die aktuellen Statusmeldungen enthält. Auch die Änderungen in den Feldern des Optionenfensters werden asynchron an den Server gesendet, sobald die Änderung am Formularelement beendet wurde. Der Server speichert die Änderungen und sendet HTML kombiniert mit JavaScript-Code zurück.

Der Umsetzungsaufwand

Während der Einarbeitungsphase musste sehr viel in der Dokumentation des Frameworks nachgelesen werden. Der eigentliche Arbeitsaufwand stand deswegen nicht direkt im Verhältnis mit dem umzusetzenden Inhalt. Nachdem die technischen Grundlagen gelegt und das Grundgerüst fertig gestellt waren, konnte ein relativ gleichmäßiger Zeitaufwand verzeichnet werden. In Tabelle 6.2 wird im oberen Teil der Arbeitsaufwand der umgesetzten Bereiche aufgelistet. Die letzte Spalte beschreibt dabei wie groß der Anteil an Anwendungslogik des jeweiligen Bereiches ist. Das Wissen über bereits umgesetzte Anwendungsbereiche erlaubt eine grobe Abschätzung über die noch umzusetzenden Teile der Anwendung, was im unteren Teil der Tabelle dargestellt wird.

Bereich	Aufwand	Anteil an Logik
NetworkIF		
Übersicht	0.5 Tage	gering
Ändern	1 Tag	hoch
IPSec		
Übersicht (mit Speichern)	1 Tag	mittel
Hinzufügen	0.5 Tage	hoch
Ändern	0.5 Tage	gering

Noch umzusetzende Teilbereiche		
Administration		
Zugang	1 Tag	mittel
Logging & eMail	1.5 Tage	hoch
Systemverwaltung	2 Tage	hoch
Einstellungen sichern	0.5 Tage	gering
iSCSI		
Übersicht (über die Targets)	0.5 Tage	mittel
iSNS-Server	0.5 Tage	mittel
Discovery	0.5 Tage	mittel
Statistik	2 Tage	hoch
Tests und Fehlersuche	6 Tage	hoch
eventuelle Erweiterungen	4 Tage	hoch
Gesamter Zeitaufwand	18.5 Tage	

Tabelle 6.2: Aufwandsabschätzung für die Implementation

6.5 Graphische Überarbeitung

Aufgrund des zeitlichen Aufwandes einer graphischen Komplettumstellung wurde die Designsuche auf eine spätere Phase verschoben. Das vorhandene Design sollte weiter verwendet und im Rahmen der inhaltlichen Veränderungen erweitert werden.

Die hierarchische Navigation

Für jede Navigationsebene existieren eine Baumnavigation im Navigationsrahmen und rechts davon detaillierten Navigationselemente im Haupt-Inhaltsrahmen. Mit der Baumnavigation kann der Benutzer schnell in tiefere Ebenen springen, ohne den gesamten Navigationspfad zu durchlaufen. Bei der detaillierten Navigation erhält der Benutzer dafür Zusatzinformationen über Systemteile. Abbildung 6.10 zeigt im linken Teil die Baumnavigation.

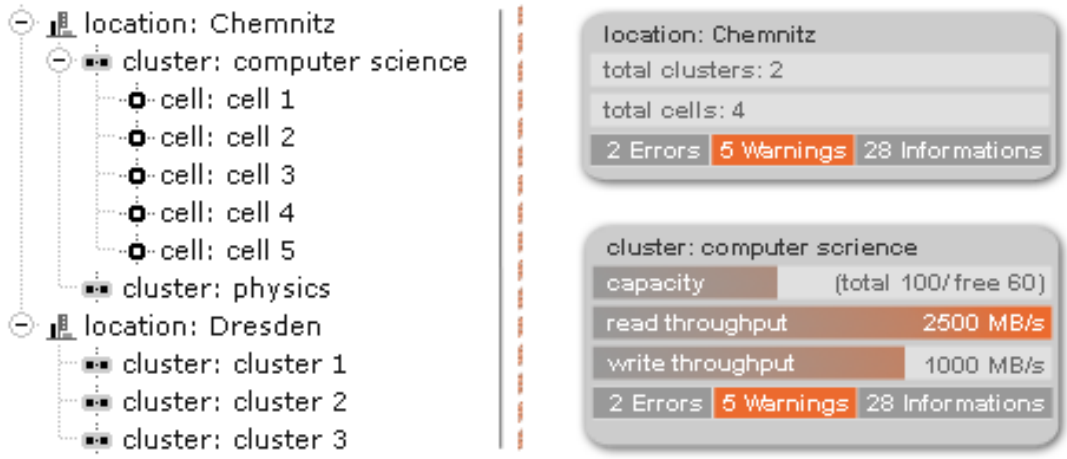

Abbildung 6.10: Baumnavigation und graphische Navigationselemente

Das Standortelement rechts oben fasst die wichtigsten Eigenschaften eines Standortes zusammen. Dies sind der Standortname, die Anzahl an eingebundenen Clustern und die Anzahl eingebundener Zellen. Die unterste Zeile des Rahmens bezieht sich auf die Logging-Informationen des Gesamtsystems, die zyklisch aktualisiert wird. Durch einen Klick auf einen der drei Knöpfe öffnet sich das Statusfenster. In ähnlicher Weise ist das Cluster-Element rechts unten im Bild aufgebaut. Es gibt Informationen über den Clusternamen, die Kapazität des freien und gesamten Speichers, den Lese- und Schreibdurchsatz des Systems und die Systemereignisse als auswählbare Knöpfe. Die beiden Striche in der Mitte stellen eine Kommunikationsbeziehung zwischen zwei Elementen einer Ebene dar. Der durchgezogene Strich steht für die Zugehörigkeit zu dem Gesamtsystem. Die gestrichelte rote Linie symbolisiert eine Backup- oder Replikationsbeziehung zwischen den jeweiligen Teilsystemen, die bei Aktivität mit einem Pfeil in die Ausführungsrichtung dargestellt wird.

Der Inhalt einer Zelle ist abhängig von der Hardwarekonfiguration des Storagesystems. In Abbildung 6.11 wird die derzeit mögliche Maximalkonfiguration des XAS1000-Systems dargestellt.

Im oberen Teil des Bildes befinden sich die zwei Controller, die über eine *Heartbeat*-System miteinander verbunden sind. Der fehlerfreie Betrieb wird über zwei bewegende Herzen symbolisiert, die bei einer Unterbrechung der Verbindung stillstehen und grau werden. Des Weiteren besitzt jeder Controller vier SAS Ports für den Anschluss von JBODs. In einem JBOD-Strang können drei JBODs in Reihe geschaltet werden, die mit jedem Controller verbunden sind. Dadurch wird die Redundanz realisiert. Die Farben für Kabel und Festplattenschächte verdeutlichen den aktuellen Systemstatus. Grau steht für „bereit“, rot für „ausgefallen“. Die dunkelgrauen, nach innen gewölbten Festplattenschächte sollen dem Benutzer zeigen, dass an dieser Stelle kein Festplattenlaufwerk in Betrieb ist. Durch das Überfahren der Hardwareelemente mit der Maus soll der Benutzer detaillierte Informationen über den Betriebsstatus bekommen, die im rechts daneben befindlichen Rahmen ausgegeben werden.

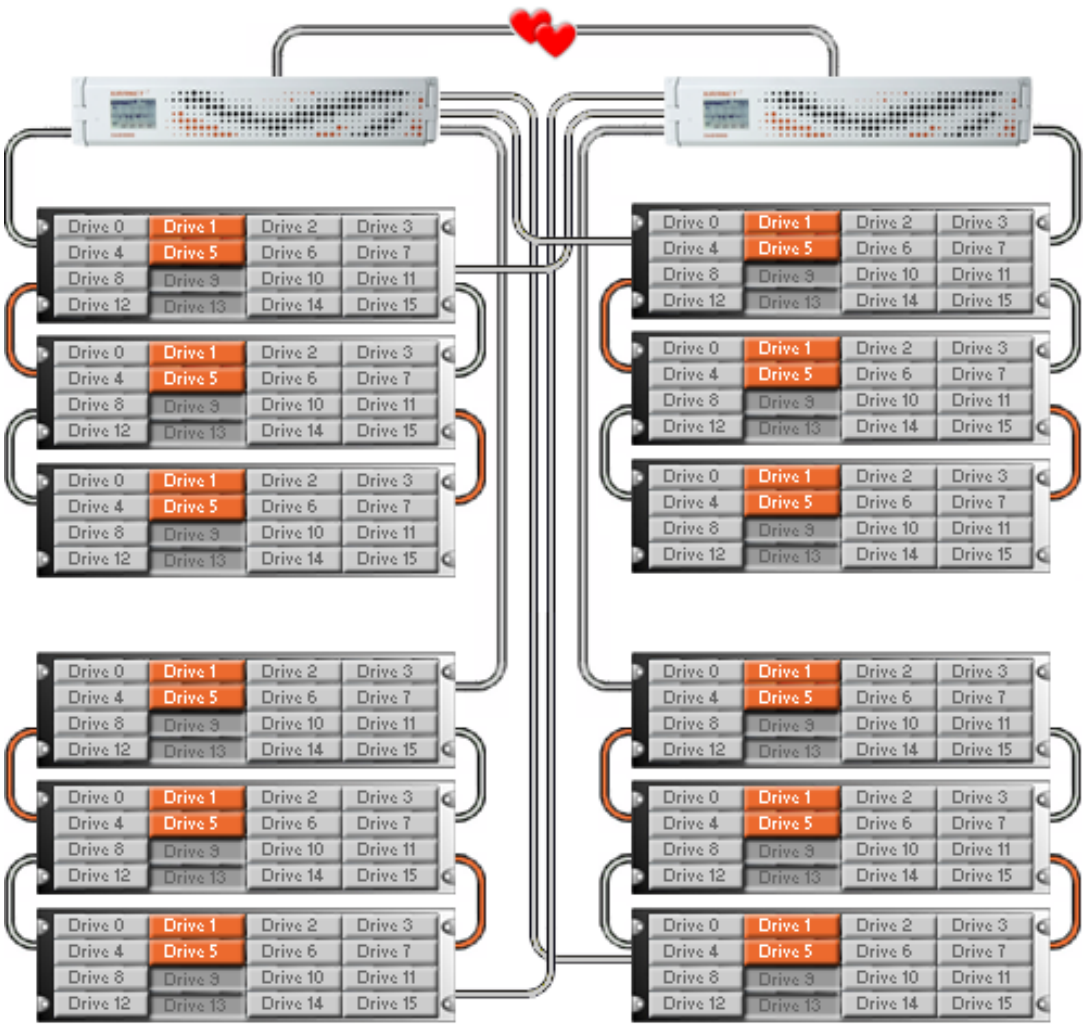
Drive 0 Drive 1 Drive 2 Drive 3
Drive 4 Drive 5 Drive 6 Drive 7
Drive 8 Drive 9 Drive 10 Drive 11
Drive 12 Drive 13 Drive 14 Drive 15

Abbildung 6.11: Die graphische Darstellung des Inhaltes einer Zelle

Die Statusleiste

In einigen der vorhergehenden Abbildungen der Managementoberfläche ist die Statusleiste bereits zu sehen gewesen. Wegen der Übersichtlichkeit ist sie permanent am unteren Fensterrand verankert und kann nicht verschoben werden. Sie liegt eine Ebene höher als alle anderen darzustellenden Inhalte der Seite und ist deshalb immer im Vordergrund.

Die Leiste selbst ist ein 21 Pixel hoher Streifen, der sich automatisch zentriert. Auf ihm sind mittig die Schalter für die Filterfunktionen positioniert. Rechts daneben ist der Copyright-Vermerk und ein Freiraum für den Ladestatus zu finden, und rechts daneben die Knöpfe für die Steuerung der ausfahrbaren Statuselemente. Der Schraubenschlüssel steht für das Einstellungsfenster, das weiße Viereck für das vollständige Ausfahren des Meldungsfensters über die gesamte Höhe und der grüne Pfeil für das teilweise Ausfahren des Meldungsfensters in die jeweilige Pfeilrichtung. Abbildung 6.12 zeigt das teilweise ausgefahrene Meldungsfenster. Dieses schiebt sich unter der Statusleiste von unten nach oben

in den sichtbaren Bereich und hat dieselbe Breite wie der äußere Rahmen der Anwendung. Bei voll ausgefahrenem Meldungsfenster, wie es in Abbildung 6.14 zu sehen ist, wird die gesamte Anwendung bündig überdeckt. Das Optionenfenster ist auf der rechten Seite der Statusleiste positioniert und fährt ca. 150 Pixel nach oben. Wie es ausgefahren aussieht und welche Formularelemente darin zu finden sind, zeigt die Abbildung 6.13. Jede Fensterbewegung ist eine gedämpfte Bewegung, was der Anwendung einen seriösen Eindruck verschafft.

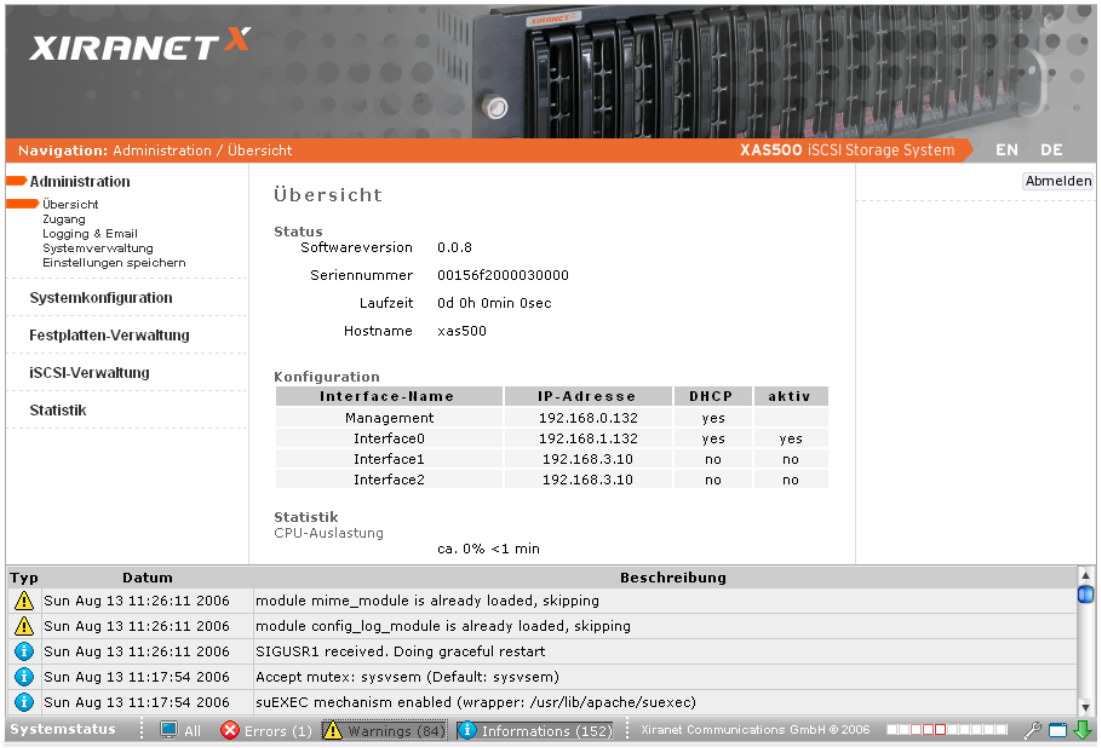

Abbildung 6.12: Die Statusleiste mit teilweise ausgefahrenem Meldungsfenster

Typ | Datum | Beschreibung
Sun Aug 13 11:26:11 2006 SIGUSR1 received. Doing graceful restart
Sun Aug 13 11:17:54 2006 Accept mutex: sysvsem (Default: sysvsem)
Sun Aug 13 11:17:54 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Sun Aug 13 11:17:54 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26
Options
show table head
highlight new event
show last 4 lines in small window
show last 300 events from log file
Systemstatus
All
Errors (1)
Warnings (84)
Informations (152)
Xiranet Communications GmbH © 2006

Abbildung 6.13: Das Einstellungsfenster für die Konfiguration der Statusleiste

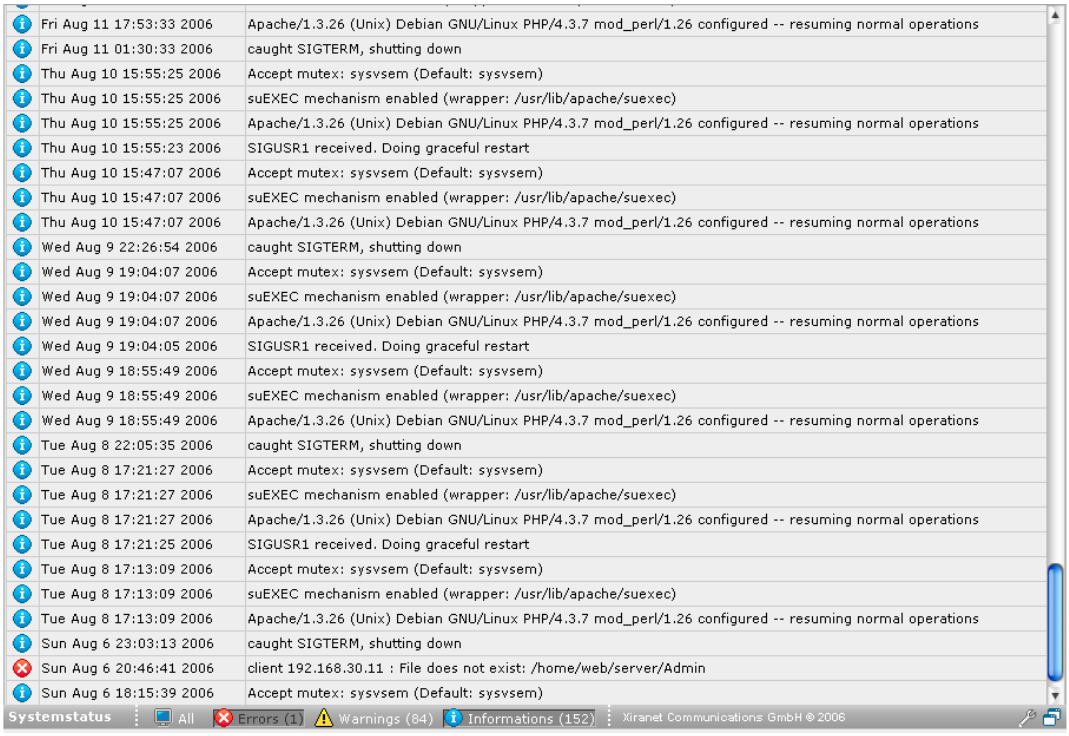
Fri Aug 11 17:53:33 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Fri Aug 11 01:30:33 2006 caught SIGTERM, shutting down
Thu Aug 10 15:55:25 2006 Accept mutex: sysvsem (Default: sysvsem)
Thu Aug 10 15:55:25 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Thu Aug 10 15:55:25 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Thu Aug 10 15:55:23 2006 SIGUSR1 received. Doing graceful restart
Thu Aug 10 15:47:07 2006 Accept mutex: sysvsem (Default: sysvsem)
Thu Aug 10 15:47:07 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Thu Aug 10 15:47:07 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Wed Aug 9 22:26:54 2006 caught SIGTERM, shutting down
Wed Aug 9 19:04:07 2006 Accept mutex: sysvsem (Default: sysvsem)
Wed Aug 9 19:04:07 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Wed Aug 9 19:04:07 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Wed Aug 9 19:04:05 2006 SIGUSR1 received. Doing graceful restart
Wed Aug 9 18:55:49 2006 Accept mutex: sysvsem (Default: sysvsem)
Wed Aug 9 18:55:49 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Wed Aug 9 18:55:49 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Tue Aug 8 22:05:35 2006 caught SIGTERM, shutting down
Tue Aug 8 17:21:27 2006 Accept mutex: sysvsem (Default: sysvsem)
Tue Aug 8 17:21:27 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Tue Aug 8 17:21:27 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Tue Aug 8 17:21:25 2006 SIGUSR1 received. Doing graceful restart
Tue Aug 8 17:13:09 2006 Accept mutex: sysvsem (Default: sysvsem)
Tue Aug 8 17:13:09 2006 suEXEC mechanism enabled (wrapper: /usr/lib/apache/suexec)
Tue Aug 8 17:13:09 2006 Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.3.7 mod_perl/1.26 configured -- resuming normal operations
Sun Aug 6 23:03:13 2006 caught SIGTERM, shutting down
Sun Aug 6 20:46:41 2006 client 192.168.30.11 : File does not exist: /home/web/server/Admin
Sun Aug 6 18:15:39 2006 Accept mutex: sysvsem (Default: sysvsem)
Systemstatus All Errors (1) Warnings (84) Informations (152) Xiranet Communications GmbH © 2006

Abbildung 6.14: Statusleiste mit vollständig ausgefahrenem Meldungsfenster

6.6 Ausblick für zukünftige Erweiterungen

6.6.1 Assistenten

In der Konzeption wurde auch der Einsatz von Assistenten geplant. Da die Implementierung dieser auf technischer Ebene sehr umfangreich ist, wurde nur eine graphische Umsetzung vorgenommen. Abbildung 6.15 zeigt das Assistentenfenster mit möglichen Konfigurationsinhalten für eine Schnellkonfiguration.

Die im Fensterkopf befindlichen Knöpfe stellen die Navigation dar und zeigen die aktuell gewählte Konfigurationsseite mit Namen in roter Farbe. Dieser Dialog kann beliebig erweitert werden. Wenn in einer späteren Version dieses Managementsystems die technische Grundlage dafür geschaffen wurde, kann dieser Assistent auf lokaler Navigationsebene eingesetzt werden. Auf globaler Navigationsebene könnte er die Zugehörigkeit eines Speichersystems zu anderen konfigurieren.

6.6.2 Weltkartenmetapher

Als weiteren Schritt der Entwicklung des neuen Managementsystems war eine komplette graphische Überarbeitung der Benutzerschnittstelle geplant. Dabei wurde die Weltkartenmetapher geprägt, in der ein Benutzer nach dem Login die Weltkarte als Navigationskomponente angezeigt bekommt. Jedes eingebundene Storagesystem wird durch einen Punkt auf der Karte dargestellt und gibt den aktuellen Status wieder. Abbildung 6.16 zeigt einen graphischen Prototypen mit der Gesamtansicht des Einstiegspunktes, die oberste Navigationsebene nach dem Login. Ein grüner Punkt verdeutlicht ein einsatzbereites System, ein ro-

Abbildung 6.15: Beispiel einer möglichen Darstellung der Assistenten

ter Punkt ein fehlerhaftes. Im Fehlerfall wird der graue Pfeil mit der Standortbeschreibung durch ein darunter liegendes Fenster erweitert. Darin befinden sich Informationen über den Namen des Fehler verursachenden Systems mit der dazugehörigen Fehlerbeschreibung. Im fehlerfreien Betriebszustand wird der Pfeil mit Informationsfenster erst dann sichtbar, wenn ein Klick auf den Punkt getätigt wurde. Mit einem weiteren Klick auf den Pfeil wird die nächst tiefere Navigationsebene, die Clusteransicht, erreicht. Unterhalb der Weltkarte befindet sich die Ebenennavigation, mit der der Benutzer direkt in die gewünschte Ebene springen kann. Das dunkelgrau eingefärbte Ebenenelement soll die aktuell ausgewählte Ebene darstellen. Rechts von den Navigationspfeilen befindet sich die Steuerung der Zoomfunktion, mit der die Kartenansicht vergrößert werden kann. Der orange eingefärbte Balken zeigt die gerade ausgewählte Zoomstufe an. Unterhalb dieser Navigationselemente befindet sich Platz für Zusatzinformationen, die in niederen Ebenen angezeigt werden müssen.

Durch diese Metapher bekommt der Benutzer einen geordneten Überblick über weltweit verteilte Systeme, die er zentral verwalten kann.

6.6.3 Replikation und Backup

Derzeit werden Pläne einer Replikationsverwaltung und eines Backupmanagementsystems entworfen, die ebenfalls mit Symfony entwickelt und in das Managementsystem integriert werden sollen.

Abbildung 6.16: Der graphische Prototype der Weltkartenmetapher

7 Zusammenfassung

7.1 Bewertung der Neukonzeption

Im Rahmen der Neukonzeption wurden viele Veränderungen vorgenommen. Die größte Änderung ist die Umstellung auf ein Open-Source-Framework, welches neue technische Funktionalitäten zur Verfügung stellt. Das alte System basierte auf einem sehr spezialisierten, proprietären Framework, welches unflexibel im Bezug auf Erweiterungsmöglichkeiten war. Das neu eingesetzte Symfony Framework bietet eine stabile Codebasis für die Weiterentwicklung der Webanwendung. Mit der Nutzung von PHP-Templates wird die Aktualisierung vereinfacht und es gibt eine klare Trennung von Anwendungslogik und Darstellungslogik. Durch die vollständige Integration der AJAX-Technologien konnte auch der Statusbalken realisiert werden, der unabhängig von den Seitenaufrufen mit dem Webserver kommunizieren kann. Er bietet dem Benutzer eine übersichtliche Darstellung der Systemereignisse mit einer entsprechenden Filterfunktion, was im alten Managementsystem schlecht umgesetzt wurde. Die größte sichtbare Neuerung im Managementsystem ist die Einführung der globalen Navigationsebene. Dadurch ist der Grundstein für die Gruppenverwaltung gelegt worden. Natürlich bringt das neue System auch einige Änderungen mit sich. Zum einen muss PHP von der Version 4.3 auf die Version 5.1 erneuert werden. Zum anderen hat sich die Größe des Frameworks von 1.7MB auf 6.5MB erhöht, was allerdings noch ein verträgliches Maß darstellt.

Die Neukonzeption war ein entscheidender Entwicklungsschritt nach vorn, der eine Vielzahl an Vorteilen mit sich gebracht hat. Das dabei verwendete Symfony Framework besitzt noch viel Potential für die Erweiterung der Webanwendung und Entwicklung neuer Programmmodule.

7.2 Ausblick

Die Entwicklung von Storagesystemen wird von jeder Firma individuell durchgeführt. Alle Hardwareelemente werden gezielt ausgesucht und die Software wird an die Anforderungen des jeweiligen Systems angepasst. Die dabei erstellten Entwicklungsstandards werden aus strategischen Gründen nicht die Firmengrenzen überschreiten. Es ist um Vieles einfacher ein eigenes System zu entwickeln als es mit anderen Herstellern abzugleichen. Des Weiteren hat ein System einen großen Vorteil, wenn es mehr Funktionalitäten anbieten kann als die Konkurrenz. Die Entwicklergruppe des Eclipse-Aperi-Projektes versucht beides zu realisieren. Einerseits soll eine Brücke zwischen den Systemen verschiedener Hersteller geschaffen werden. Andererseits soll jeder Hersteller sein individuelles Produkt gestalten können und ebenfalls den Vorteil der erhöhten Flexibilität nutzen dürfen.

Aperi ist ein Open-Source-Framework, das sich an Standards orientiert und erweiterbar ist. Es soll die Infrastruktur, die ein Nutzer zur Verwaltung seiner Speicherhardware benötigt, stark vereinfachen. Die Aperi-Projektgruppe wurde im Juni 2006 ins Leben gerufen und begann bereits Anfang September mit der Implementierung. Ziel der Gruppe ist es, in enger Zusammenarbeit mit der *Storage Networking Industry Association*[1] (SNIA) ein Framework zu erschaffen, das Interoperabilität und weniger Komplexität in Speicherumgebungen zur Verfügung stellt. Die 10 weltweit führenden Unternehmen auf diesem Gebiet sind in der SNIA aktiv und arbeiten an Aperi mit, um einen einheitlichen Standard, die *Storage Management Initiative Specification* (SMI-S) zu realisieren. Das Projekt soll die Hindernisse zwischen den Systemen der Marktführer abbauen und die Entwicklung der Managementsoftware vereinfachen. Der Endnutzer hat den Vorteil der höheren Flexibilität und größeren Auswahl an Storagesystemen, die miteinander kombiniert werden können, und dadurch die Unterhaltungskosten im Unternehmen senken. Er hätte dann die Möglichkeit, alle Storagsysteme mit diesem Standard durch nur eine Verwaltungssoftware zu konfigurieren.

Die folgenden Schwerpunkte soll das Framework umsetzen:

- Ermittlung und Darstellung von Ressourcen
- Abbildung von Topologien
- Verwaltung von Festplatten und Bandspeichermedien
- Hardwarekonfiguration und LUN-Zuweisung
- Verwaltung der SAN Struktur
- Grundlegende Geräteverwaltung
- Graphische Benutzerschnittstelle

Das Projekt befindet sich gegenwärtig in der Erstellungsphase. Diese umfasst die typischen Schritte der Softwareentwicklung bis hin zu der Recherche über die Verwendungsrechte des eingeplanten geistigen Eigentums. Dazu gehört außerdem das Erzeugen der Community, die von der Projektgruppe in drei Gruppen unterteilt werden.

- **Committers** - Die aktive eingebundene Nutzergemeinde, die im Wesentlichen dazu beiträgt, dass das Projekt bekannt wird und durch ihre Unterstützung am Leben gehalten wird. Dazu gehören auch Verbesserungsvorschläge oder Programmieren von Softwarekomponenten.
- **Users** - Die normale aktive Nutzergemeinde, die ein Open-Source-Projekt bezüglich seiner Nützlichkeit bewertet und den Entwicklern eine Rückmeldung gibt.
- **Plugin Developers** - Diese Gruppe erweitert ein Framework und zeigt damit, dass die Eclipse-Projekte durch gut dokumentierte Anwendungsschnittstellen vielseitig nutzbar sind. Auch das ist ein Maß für den Erfolg eines Open-Source-Projektes.

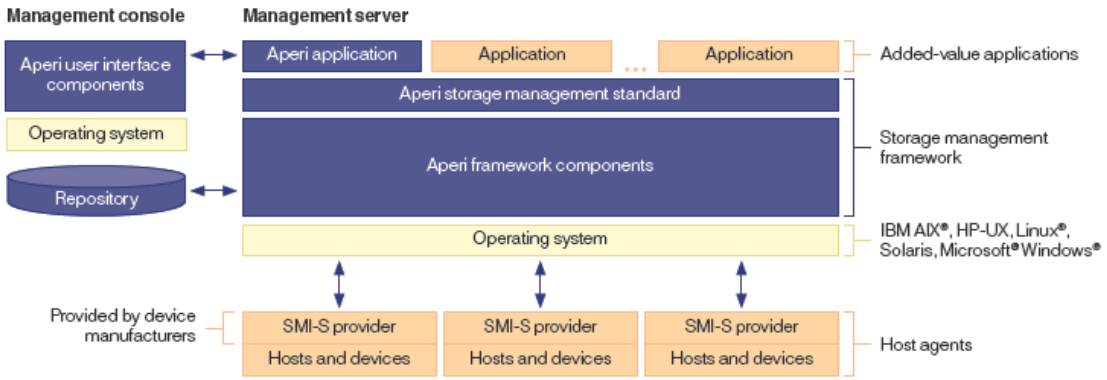

Abbildung 7.1: Die Architektur des Aperi Framework (Quelle: www.eclipse.com/aperi)

Die Architektur des Frameworks, wie sie in Abbildung 7.1 zu sehen ist, teilt sich in die Grundelemente Managementserver, Managementkonsole, Host Agent und Repository auf. Der Hostagent implementiert Schnittstellen nach dem SMI-S Standard, um die Hardware für den Managementserver bereit zu stellen. Der SMI-S Standard basiert auf den Standards CIM und WBEM, die für die Kommunikation eingesetzt werden. Speziell das Transportprotokoll WBEM ist unabhängig von Programmiersprache, Plattform und Compiler. Die eigentliche Übertragung wird vom CIM-XML Protokoll basierend auf HTTP übernommen. Andere Standards für die Übertragung, wie beispielsweise Web Services und SSL, sind ebenfalls in dem Framework integriert. Der Managementserver erweitert die Eclipse Plattform mit Funktionalitäten des Storagemanagements. Die Aperi-Anwendungen können auch beliebig durch eigene Anwendungen erweitert werden. Aperi stellt auch ein graphische Benutzeroberfläche für den Zugriff auf den Managementserver zur Verfügung. Diese GUI ist dabei nicht an das Betriebssystem des Managementservers gebunden.

Der Ansatz ist im Bezug auf die Interoperabilität ein großer Fortschritt. Der Einsatz im Xiranet-Managementsystem ist allerdings mit einem hohen Erweiterungsaufwand verbunden, da im Management-Deamon eine zweite, SMI-C konforme Schnittstelle implementiert werden müsste. Alle in Abbildung 7.1 rosa eingezeichneten Komponenten müssten dann von Xiranet realisiert werden. Die GUI nutzt eine Eclipse Rich Client Plattform als Anwendung und ist damit an ein Betriebssystem mit Java Laufzeitumgebung gebunden. Damit würde das Managementsystem die Eigenschaft der Plattformunabhängigkeit durch den Einsatz von Webbrowsern verlieren.

[1]Die SNIA ist eine nicht am Profit orientierte Handelsvereinigung, die in Form von technischen Arbeitsgruppen Speichernetzwerke komplettieren und verbessern wollen.

Literaturverzeichnis

[1] Hufferd, J.L.: *iSCSI, The Universal Storage Connection*. Addison-Wesley, **2002**, pp. 1–11.

[2] Schnabel, P.: Fibre channel (fc). *Tech. Rep.*, das Elektro-Kompendium, http://www.elektronik-kompendium.de/sites/net/0905071.htm, **2006**.

[3] Clark, T.: *IP SANs*. Addison-Wesley, **2001**, pp. 193–199.

[4] Seifert, E.: Einführung in zope. *Tech. Rep.*, DZUG - Deutschsprachige Zope User Group, http://www.zope.de/dokumentation/einfuehrung/zopebuch01, **2006**.

[5] Langner, T.: *Verteilte Anwendungen mit Java*. Markt+Technik Verlag, **2002**.

[6] O'Reilly, T.: What is web 2.0. *Tech. Rep.*, O'Reilly Verlag, http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html, **2005**.

[7] Britannica corporate site. *Tech. Rep.*, Encyclopædia Britannica, Inc., http://corporate.britannica.com/about/, **2006**.

[8] Hofmann, O.: Komposition von web services. *Tech. Rep.*, TU Dresden, **2005**.

[9] Jones, R.: A restructuredtext primer. *Tech. Rep.*, DocUtils, http://docutils.sourceforge.net/docs/user/rst/quickstart.html, **2005**.

[10] Kicker, E.: Einführung in json - javascript object notation. *Tech. Rep.*, very-clever.com, http://www.very-clever.com/json.php, **2005**.

[11] Garrett, J.J.: Ajax: A new approach to web applications. *Tech. Rep.*, adaptive path, http://www.adaptivepath.com/publications/essays/archives/000385.php, **2005**.

[12] Frank Manola, E.M.: Rdf primer. *Tech. Rep.*, W3C, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, **2004**.

[13] Developers, A.: *Atom Syndication Format - Introduction*. http://www.atomenabled.org/developers/syndication/, **2004**.

[14] Sixtus, M.: Die Humanisierung des Netzes. *Die Zeit 35*.

[15] Keen, A.: Web 2.0 - the second generation of the internet has arrived. it's worse than you think. *Tech. Rep.*, News Corporation, Weekly Standard, http://weeklystandard.com/Content/Public/Articles/000/000/006/714fjczq.asp?pg=1, **2006**.

[16] Kristöfl, D.R.: Evaluation von lernplattformen: Verfahren, ergebnisse und empfehlungen(version 1.3). *Tech. Rep.*, Bundesministerium für Bildung, Wissenschaft und Kultur(BMBWK), http://www.bildung.at/statisch/bmbwk/lernplattformen_evaluation_und_ergebnisse-_1_bis_3.pdf, **2005**.

[17] Nielsen, J.: Ten usability heuristics. *Tech. Rep.*, Nielsen Norman Group, http://www.useit.com/papers/heuristic/heuristic_list.html, **1994**.

[18] Potencier, F.: *Symfony Project*. Sensio Web Agency, www.symfony-project.com, **2006**.

[19] Gell, G.: Symfony-projekt. *PHP Magazin 3* (**2006**), 60–64.

[20] Meißner, K.: *Web- und Multimedia Engineering*. Institut für Software- und Multimediatechnik, TU Dresden, **2006**.